

€ ESPRIT



COMMISSION OF THE EUROPEAN COMMUNITIES

Directorate-General
TELECOMMUNICATIONS, INFORMATION
INDUSTRIES AND INNOVATION

ESPRIT '87 Achievements and Impact

Part 2

North-Holland

ESPRIT '87

Achievements and Impact

ESPRIT '87

Achievements and Impact

Proceedings of the 4th Annual ESPRIT Conference
Brussels, September 28-29, 1987

Edited by

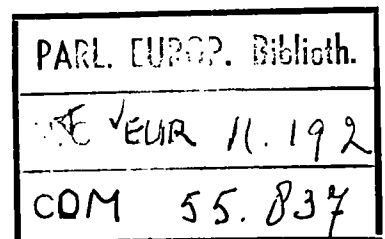
COMMISSION OF THE EUROPEAN COMMUNITIES
Directorate-General TELECOMMUNICATIONS,
INFORMATION INDUSTRIES and INNOVATION

Part 2



1987

NORTH-HOLLAND
AMSTERDAM • NEW YORK • OXFORD • TOKYO



© ECSC-EEC-EAEC, Brussels-Luxembourg, 1987

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the copyright owner.

ISBN Part 1: 0 444 70331 4

ISBN Part 2: 0 444 70332 2

ISBN Set : 0 444 70333 0

Publishers:

ELSEVIER SCIENCE PUBLISHERS B.V.

P.O. Box 1991

1000 BZ Amsterdam

The Netherlands

Sole distributors for the U.S.A. and Canada:

ELSEVIER SCIENCE PUBLISHING COMPANY, INC.

52 Vanderbilt Avenue

New York, N.Y. 10017

U.S.A.

Publication No. EUR 11192 of the
Commission of the European Communities,
Directorate-General Telecommunications,
Information Industries and Innovation,
Luxembourg

LEGAL NOTICE

Neither the Commission of the European Communities nor any person acting on behalf of the Commission is responsible for the use which might be made of the following information.

PRINTED IN THE NETHERLANDS

CONTENTS

Part 2

IV. OFFICE SYSTEMS

Human Factors for Usability Engineering (P 385) B. SHACKEL (Paper presented in Plenary Session)	1019
1. Analysis and Design Tools - Human Factors	
Basic Concepts of the Functional Analysis of Office Requirements (P 56) G. SCHAEFER and R. HIRSCHHEIM	1041
Human, Organisational and Economic (HOE) Factors in IT Uptake Processes in Complex Work Environments (P 1030) G. RYAN, K. CULLEN, R. WYNNE, T. RONAYNE, J. CULLEN and C. DOLPHIN	1053
Evaluation Criteria for the Analysis of Office Descriptions Based on Petri Nets (P 285) V. DE ANTONELLIS and C. SIMONE	1066
OSSAD - Methodology : Results of the Analysis Phase (P 285) E. BESLMUELLER, S. CASERTA, D.W. CONRATH and P. DUMAS	1077
Prototyping as an Automatic Tool for Designing Office Systems (P 813) A. KIEBACK and W. KERBER	1091
Cognitive Simulator for User-Interface Design (P 234) P. BYERLEY, P. BARNARD, D. CARR, A. FOSTER, T. FOWLER, R. SAFFIN and G. WARD	1101
2. Integration and Application	
The ProMinanD Design Cycle and Evaluation Plan (P 878) K. MÖLLENBACH and A.B. PETERSEN	1111
Aspects of Multi-Media Retrieval Shown by ESPRIT 901 (P 901) P. HILL and N. YOUNG	1124
3. Image and Speech Handling	
Photovideotex Image Compression Algorithms - Towards International Standardisation (P 563) G.P. HUDSON	1137
MIAC - ESPRIT Project 1057 for Multipoint Interactive Audiovisual Communication (P 1057) W.J. CLARK	1149
A European Project on Coding of Moving and Still Pictures at Very Low Bit Rates (P 925) J. DAVID and J.P. HAAG	1161



Text-to-Speech Synthesis in an Office Environment (P 64) D. BOILLON, H. BUETHER, J.P. LEFEVRE, L. NEBBIA and L.C.W. POLS	1175
Amorphous Silicon Contact Imager for Office and Graphic Applications (P 1051) N. KNIFFLER, J. VAN DAELE, J. NIJS, Z-M. QIAN, H. PATTYN, J. POORTMANS, E. FOGARASSY, C. FUCHS and J.C. DESOYER	1188
4. Workstations and MMI	
An Office Assistant Prototype Using a Knowledge-Based Office Model on a Personal Workstation (P 82) M. ADER and M. TUENI	1205
Representation Aspects of Knowledge-Based Office Systems (P 82) K. VAN MARCKE, V. JONCKERS and W. DAELEMANS	1226
Formatting Documents and Optimization's Problems (P 367) J.-M. VAN VYVE	1239
COCOS Architecture and its MMI Model (P 956) N. NAFFAH and M. TEXIER	1251
An Implementation of PARLOG Using High-Level Tools (P 956) J. GARCIA, M. JOURDAN and A. RIZK	1265
Types and Type-States in LE TOOL (P 956) H. BORRON	1276
Functional Models of Visual Perception in Office Conditions (P 612) D. BOSMAN	1288
Display Modelling and its Software Implementation (P 612) I. PLACENCIA PORRERO	1300
Architecture Requirements and Specifications Display Simulator (P 612) A.E. VAN DER MEULEN	1313
The Role of Paper in the Automated Office (P 295) U. BOES, W. DOSTER, G. FOGAROLI, H. LOEBL and G. MASLIN	1325
Dynamic Handwritten Script Recognition (P 295) Ph. WRIGHT	1341
Linguistic Analysis of the European Languages (P 291) V. VITTORELLI	1358
5. Documents Architecture, Storage and Retrieval (part II)	
Standardized Interchange Formats for Documents (P 121) G. KROENERT	1367
The ESPRIT PODA Demonstration of ODA at CeBIT 87 (P 1024) P.J. ROBINSON, E. KOETHER, I.R. CAMPBELL-GRANT, O.R. MOREL and F. FASANO	1378

Retrieval of Multi-Media Document Images in MULTOS (P 28) P. CONTI and F. RABITTI	1389
Office Procedures : A Formalism and Support Environment (P 231) S. BAKER, B. CAULFIELD, R. HOEKSTRA, M. LOPEZ, M.R. RATHGEB, M. SHEPPARD, G. STANDEN and T.W. TRUCKENMUELLER	1413
Techniques for Filing and Retrieval of Office Information (P 59) G. MCALPINE	1425
6. Human Factors (part I)	
HUFIT : Human Factors in Information Technology (P 385) K.-P. FAEHRICH, J. ZIEGLER and D.G. DAVIES	1443
Human Factors Engineering of Interfaces for Speech and Text in an Office Environment (P 385) F.L. VAN NES	1452
7. Distribution Systems and Communications	
The INCA Workstation (P 395) C. BATHE and H.P. GODBERSEN	1459
INCA Directory Services (P 395) S.E. KILLE	1472
The LION Project : A Status Report (P 169) A. LUVISON, G. ROULLET and F. TOFT	1477
Object-Oriented Architecture for Distributed Office Systems (P 834) C. HORN and S. KRAKOWIAK	1490
An Architecture for System and User Management Tools in a Distributed Office System (P 834) A.J. NESS, F. REIM, H. MEITNER, R.A. PERCY and S.S. MAKH	1501
Security Development : Traditions and Innovations (P 998) S. FARRELL, E. STEPHENS, F. WILLIAMS, M. FOUGERE, G. RUGGIU and V. SOREL	1517
V. COMPUTER INTEGRATED MANUFACTURING	
CNMA - Putting Standards to Work (P 955) J. BOOTH and B. GIRARD (Paper presented in Plenary Session)	1533
1. CAD and CAM Tools	
CAD Data Transfer : The Goals and Achievements of Project 322 E.G. SCHLECHTENDAHL and U. GENGENBACH	1547
Development of Advanced Modelling Techniques (P 322) H. GRABOWSKI, R. ANDERL, B. PAETZOLD and S. RUDE	1570

PAPILLON - A Support Environment for Graphical Software Development (P 496) C. CHEDGEY	1583
ACCORD : A Step Toward Integrated Computer Aided Analysis Environments (P 1062) J.P. PATUREAU	1594
Development of an Integrated Process and Operations Planning System with the Use of Interactive 3D Modelling Techniques (P 409) G. ERNST	1608
2. Knowledge Based Developments in CIM	
Expert Supervisory Control of a Flexible Manufacturing System (P 809) H. HEINZ-FISCHER, R. LOS, H. POELS, C. POHL, J. RINGEL and H. STIENEN	1619
Intelligent Models for Assembly Design, Planning and Control (P 384) S.P. SANOFF and B. DWOLATZKY	1638
Development Towards an Application Generator for Production Activity Control (P 477) COSIMA Project Team	1648
Artificial Intelligence in Production Scheduling (P 418) M. CLARK and F. FARHOODI	1662
Aspects of Knowledge-Based Factory Supervision Systems (P 932) W. MEYER and R. ISENBERG	1671
3. Robots - Developments in Manufacturing Systems	
VLSI and Real Time Control of Plant Automation Systems (P 179) P.R. DOREE	1693
An Integrated Sensor-Based Robot System Using Tactile and Visual Sensing (P 278) N. GHANI, Z.G. RZEPCZYNSKI, P.M. HAGE and A.E. ADAMS	1707
Planning and Programming of Robot Integrated Production Cells (P623) G. SPUR et al.	1716
4. Evaluation of CIM Systems	
C-BAT : A Methodology and Software Toolkit to Help Small- to Medium Sized Firms to Assess CIM (P 909) M. KORIBA and P.K. EYRES	1745
Experimental Center for System Integration in CIM (P 812) C. BORASIO, P. MURCHIO, R. GROPPETTI and D. SOLA	1765
CIM-OSA Strategic and Management Issues (P 688) B. LORIMY and P. RUSSELL	1777

5. Human Factors (part II)

Human Skill and Computer Power (P 1199)
A. AINGER and S. MURPHY 1795

VI. INFORMATION EXCHANGE SYSTEMS**1. Network Infrastructure : Development Projects**

The OSI-PC : An Introduction (P 718)
D.H. SOMMER 1807

CACTUS, An X400 for Private Management Domains (P 718)
R.D. KILLICK 1816

Implementation of a Network Administration Conform to the OSI
Architecture in the ESPRIT Information Exchange System (P 33)
W. BLUMANN, J. COLASSE, T. COOK, C. REMY, G. THOMAS and B. WINTER 1822

The ESPRIT Directory System (P 719)
F. SIROVICH 1838

INDEX OF AUTHORS 1849

PROJECT ACRONYM INDEX 1853

KEYWORD INDEX 1855

IV. OFFICE SYSTEMS

Paper presented in the Plenary Sessions:

Human Factors for Usability Engineering - P 385 1019

Parallel Sessions

- | | |
|---|-------------|
| 1. Analysis and Design Tools - Human Factors | 1041 |
| 2. Integration and Application | 1111 |
| 3. Image and Speech Handling | 1137 |
| 4. Workstations and MMI | 1205 |
| 5. Documents Architecture,
Storage and Retrieval (Part II) | 1367 |
| 6. Human Factors (Part I) | 1443 |
| 7. Distributed Systems and Communications | 1459 |

Project No. 385

HUMAN FACTORS FOR USABILITY ENGINEERING

B. Shackel

HUSAT Research Centre, Department of Human Sciences,
University of Technology, Loughborough, LE11 3TU, U.K.

It is now accepted more widely in the IT industry that usability must be given as much care and attention as functionality. Within the ESPRIT Office Systems Programme there is a substantial Type A Project to focus attention and work in this broad area - No. 385 HUFIT (Human Factors for Information Technology).

This paper reviews both the HUFIT Project and more generally the contributions available from Human Factors. The importance of usability is emphasised. The potential benefits from using Human Factors are illustrated. Finally, the aims, activities and progress of the HUFIT project are presented in outline.

1. INTRODUCTION

In his opening address to the First ESPRIT Technical Week, Carpentier (1984) said "let me remind you of the strategic objective. It was - 'The achievement of technological parity with, if not superiority over, world competitors within ten years' - I do not for a moment believe that European technology, in the sense of our ability to innovate and produce new ideas, is in any way inferior; but there is no doubt that Europe has fallen behind in its application of technology . . . "

This paper is also concerned with a technology pioneered by scientists and researchers in Europe, but in which Europe is beginning to fall behind in its application to the field of information technology. The discipline concerned, which will become crucial for the IT industry during the next ten years as 'technology-push' is replaced by 'market-pull' driven by purchasers and users, is the discipline of Ergonomics or Human Factors.

In a survey report prepared in 1984 for the Commission of the European Communities (Shackel 1985a) I wrote "while research in Europe has been in the lead in the past, and may still lead in certain areas of Information Technology Ergonomics, this situation is changing rapidly. Moreover, in the area which is generally accepted now to be the most important, that of software ergonomics, the U.S.A. is probably in the lead - certainly in quantity of research. The biggest difference between Europe and the U.S.A. is the much greater development already of Human Factors in industry in the U.S.A". While quite a number of research projects in Ergonomics have been funded under the ESPRIT programme (and also, for example, within the Alvey Programme in Britain and similarly in other community countries), the application of this technology has grown very little in the European IT industry in the last three years.

No doubt the faster and greater exploitation of research in the U.S. industry is helped by more venture capital available, more favourable tax rates, a more entrepreneurial economic climate and a larger home market. On the other hand, business management in the U.S. appears to be even more demanding of proof of commercial pay-off; within such a cost-conscious framework it is

noteworthy that in a short survey of 15 large U.S. computer companies in 1984 I received statements of growth in Human Factors staff from twofold to fivefold with an average of threefold (ie 300% growth).

Therefore, this paper aims to show to the European IT industry in ESPRIT both the importance of Human Factors Technology and the size of the potential benefits from using Human Factors. After a brief historical review of Human Factors in relation to computing and IT, the importance of Human Factors in the design and use of IT systems will be described. The benefits from using Human Factors will be illustrated with several specific examples. Finally, the aims and progress of ESPRIT Type A Project No. 385, Human Factors for Information Technology (HUFIT) will be reviewed.

2. HISTORICAL CHANGES AND THEIR CONSEQUENCES FOR USABILITY

The reason for this section is to emphasise the rapid growth and change in computing which has fundamentally changed the predominant type of users and their expectations. For a fuller review see Gaines (1984), and for useful references see Burch (1984) and Shackel (1985a).

At the beginning of the digital computer era, the designers of computers were specialists and the users of computers had to become computer specialists. The potential power of this new machine and the speed of computation was so useful for certain scientific disciplines that some scientists found it worth the cost of time and effort to learn how to use it.

In the late 1950's the potential for the computer in industry and commerce was recognised, and the first serious business machines were developed; again, they were designed by computer specialists for use by data processing professionals. From the early 1970's the mini-computer and remote terminal access to the time-sharing mainframe brought computer usage nearer to the layman. However, already the difficulties for the non-specialist and the problems of human-computer interaction were recognised (Nickerson 1969, Shackel 1969, Sackman 1970).

The advent of the micro-computer, in widespread use from 1980, has already caused great growth in the use of computing for many different purposes by non-specialists of all types - from bank clerk to business executive, from librarian to life insurance salesman and from secretary to stockbroker and space traveller. This rapid growth in computing, leading to widespread usability problems, is summarised in Figure 1.

The result of this rapid growth is that both the market for the IT industry and the users of IT equipment are changing very rapidly. The market is becoming much more selective, partly through experiences of poor usability. The users are no longer mainly computer professionals, but are mostly discretionary users (Bennett 1979). As a result, the designers are no longer typical of or equivalent to users; but the designers may not realise how unique and therefore how unrepresentative they are.

Moreover, with the growth of IT, the many new users bring different needs to be satisfied. Earlier users were committed to using computers either because of personal interest or job requirements. But the potential new users are such people as managers, physicians, lawyers and scientists who are committed to their tasks but not at all to the computer. They have choice and will only use computers if they are appropriate, useful and usable. So the market now contains important new categories of users. Moreover, some predict that these end users will be the primary decision makers in the future about acquiring equipment. Thus to be successful, the IT industry must improve the usability of interactive systems, and to do so the

understandable orientation of designers in the early years must now be completely reversed; designing must start with the end users and be user-centred around them. Therefore the Human Factors aspects become paramount.

Computer type	Approx growth era	Main Users	Issues
Research machines	1950s	Mathematicians Scientists	Size, reliability, users must learn to do every bit of programming
Mainframes	1960s & 1970s	Data-processing professionals supplying a service	Users of the output (business managers) grow disenchanted with delays, costs, lack of flexibility
Minicomputers	1970s	Engineering and other non-computer professionals	Users must still do much programming; usability becomes a problem
Microcomputers (plus applications packages)	1980s	Almost anyone	Therefore usability is the major problem

Figure 1. Growth of digital computers and user issues

However, the growth of attention to the consequential human factors and usability aspects was slow to develop. Through the 1960s such work as existed was scattered and mostly, in the USA especially, related to military systems. Through the 1970s significant work developed though still largely in small, somewhat isolated groups. But in 1980 the recent considerable growth was crystallized in four books (one from a conference), to be followed by several books each year thereafter, by a second main journal in 1982 and by seven major conferences in 1982-84 - see Figure 2.

1959	1st recorded paper in the literature (Shackel 1959) as reported by Gaines 1984
1969	1st major conference ('International Symposium on Man-Machine Systems')
1969	'International Journal of Man-Machine Studies' started
1970	Foundation of HUSAT Research Centre, Loughborough University
1970-73	4 seminal books published (Sackman, Weinberg, Winograd, Martin)
1976	NATO Advanced Study Institute on 'Man-Computer Interaction'
1980	Conference and book on 'Ergonomics Aspects of Visual Display Terminals' (Grandjean & Vigliani 1980)
	Three others books (Cakir et al, Damodaran et al, Smith & Green)
1982	Journal 'Behaviour and Information Technology' started
1982 to 1984	7 major conferences held in USA, UK and Europe with attendances ranging from 180 to over 1000 with an average of 485

1983	ESPRIT begins
1985	Journal 'Human-Computer Interaction' started
1985	ESPRIT HUFIT Project No. 385 begins 1 December

Figure 2. Growth of attention to ergonomic aspects of human-computer interaction (see Gaines 1984 for a review)

3. THE IMPORTANCE OF HUMAN FACTORS AND USABILITY

There is now no doubt about the importance of human factors in the eyes of the computer industry in the USA, where there is greater development of human factors in industry than in Europe. This is particularly evident in the large numbers of human factors professionals and others from industry at the regular ACM CHI (Computer Human Interaction) conferences; the total attendances in 1983, '85, '86 and '87 have been 1000, 1200, 1300 and 1400, and the percentage from industry has been 70% to 80% each time. Ironically, this rapid growth in attention to human factors in the U.S. industry is attributable, at least in part, to a European ergonomic standard, namely the German DIN standard for keyboard height to be not more than 30 mm. The realisation that an ergonomic standard could override all other aspects in the marketplace came as a big surprise and had a powerful effect on quite a number of U.S. companies.

To illustrate this changed situation one merely has to note the marked change of emphasis upon human factors in IBM which was handed down from the very top (Shackel 1986a). As a result, special conferences have been held, a worldwide programme of short courses for IBM engineers was instituted, and usability became of equal importance with functionality.

The following excerpt is typical of the writings of quite a number of the ergonomists in this field some years ago, but it is taken directly from a lecture by the IBM Vice President and Chief Scientist (Branscomb 1983).

"All that has changed. No longer the exclusive tool of specialists, computers have become both commonplace and indispensable. Yet they remain harder to use than they should be. It should be no more necessary to read a 300 page book of instructions before using a computer than before driving an unfamiliar automobile. But much more research in both cognitive and computer science will be required to learn how to build computers that are that easy to use. That is why our industry is paying increasing attention to the field of applied psychology called human factors, or ergonomics.....Equally neglected has been human factors at the level of systems design. We know that system architecture has significant and widespread implications for user friendliness, but we know next to nothing about how to make fundamental architectural decisions differently, in the interest of good human factors....
...Thus the effort to design for ease of use could benefit enormously from basic research, not only in adaptive systems and computational linguistics, but above all in terms of controlled experiments involving actual use by representative end users - for you can't evaluate ease of use without use."

Finally, in Britain the case has been presented in an authoritative and well-illustrated report by the National Electronics Council (1983), in which also several case studies are described to emphasise the need to design for people and build in usability from the start.

4. POTENTIAL BENEFITS FROM USING HUMAN FACTORS

The philosophy of ergonomics is concerned with ensuring a good match of the equipment, workplace, environment and organisation to the people therein so as to promote comfort, convenience, efficiency, health and job satisfaction; it does not have productivity as its principal goal. But one must always recognise that managers and designers often need to see a potential advantage in productivity or economy to be persuaded to use ergonomics; therefore, the importance of this work can best be shown by examples of direct benefits, converted into cash terms if possible.

In the following sections I shall present a number of examples to illustrate the range and type of benefits to be expected from applying human factors knowledge and methods appropriately in the design, implementation and usage of information technology. First, to indicate the scope and range of applications already made, a number of cases will be briefly mentioned and referenced. Then, to illustrate the possibilities more fully, five examples will be described at greater length with costed benefits where possible.

4.1 AN ASSEMBLAGE OF EXAMPLES

In the early days we could only point to problems and failures which badly needed ergonomics ministrations; some recent examples of such failures have been given, for example, in section 2 of the National Electronics Council Report (1983) - of the Navy's navigators reverting to pens and paper and of doctors rejecting software - and by Chapanis (1985) - of some user problems with a recent IBM software system. Instead of this negative approach which had to be taken in the past, it is pleasing to be able to give a whole range of examples all of which are successful case studies of human factors implemented and paid for by the organisations concerned. The examples have been chosen to represent many other successful applications at the equipment level, the workplace level, the organisation level, and the larger system and national level.

At the equipment level here are four examples. The first comes from within an IT company and the other three examples are work done by ergonomists as external consultants. Bewley et al (1983) described the human factors testing which they report as "integral to the design process of the Xerox 8010 'Star' workstation". Moore & Dartnall (1982) analysed for British Gas the ergonomic issues of central heating timer/programmer units and produced a number of designs leading to a final version. Galer & Yap (1980) assisted a hospital with the design of a monitoring unit for use in an intensive care ward; a new interface design was produced and tested, and the new device was found to be better than the original and was preferred by the users to manually completed charts. Finally, Davies (1981) organised the total ergonomic design of a portable billing machine for the Plessey Company which went into regular production (see fuller description below)..

At the workplace level, because so much human factors work has been done, I shall mention only one case study. In section 6 of the National Electronics Council Report (1983) the case study is presented of the Merseyside Police Control Room in which the design was led by the ergonomists. Many other examples of human factors work for workplace design are given in Grandjean (1984) and in the special issue of Behaviour and Information Technology (Vol.3 No.4 December 1984 'Ergonomics and Design in the Electronic Office') containing the papers of the Ergodesign'84 Conference.

At the organisational level, quite a number of case studies are presented in the Symposium Proceedings edited by Hendrick & Brown (1984) and by Brown & Hendrick (1986). For example, Bowman (1986) describes the successful procedures at Honeywell for setting up cross-functional design teams, with

representatives of both worker-users and manager-users, to search jointly for solutions in developing their own internal computer systems. The case study by Shackel & Eason (1986) describes the use of pilot system prototyping to develop the specification for an online distributed system for a company with 32 branches. Finally, some of the ongoing work at HUSAT concerned with user acceptability and user support in a large national organisation is summarised by Douglas & Marquis (1985) and Marquis & Douglas (1985).

At the large system and national level the Olympic Message System (Boies et al 1985) is another good example of the human factors team taking the lead in the development of a major system where the user interface is the critical feature for success or failure; there is no doubt about the success of this system at the Los Angeles Olympic Games. In Britain the Government's Department of Health and Social Security has embarked upon a ten year computerisation programme which will link all the many social security benefit offices in cities and towns throughout the country into a large online computer network and database system. HUSAT (1983) was asked to produce a human factors strategy plan as a contribution to this programme, and one of the major projects ongoing in HUSAT is the implementation of the human factors strategy in cooperation with the many users and design staff in the DHSS.

These examples help to show the growing maturity of the human factors contribution as an essential partner in the design and development process of information technology. There are still many organisations which need to hear and absorb this message, but the more there are successful contributions like those referenced the more widely will the message be heard.

4.2 INTERFACE DESIGN - SUCCESSFUL SOFTWARE

This example comes from the NCR Corporation at Dayton, Ohio, and in their paper Keister & Galloway (1983) state that "the major reason for the study was to clearly demonstrate to software developers the benefits of improved Human Factors design. By taking a reasonably successful product and showing that it can be substantially improved by attention to Human Factors design concerns, we wanted to demonstrate:

1. Improved throughput
2. Reduced error rates
3. Easier learning
4. Better user acceptance
5. Dollar savings to customers."

They selected a reasonably successful NCR package which was already being used by more than 500 customers. Following detailed analysis, the human interface portions of two of the programs from the application were selected which involved heavy user-system interactions. Changes were made to the interfaces under the categories of:- screen formatting, screen content, elimination of abbreviations, consistency, data entry/change procedures, error correction and feedback, on-line assistance; and in various details. The original and modified versions of the programs were then compared using representative data entry operators.

The comparison experiments were properly controlled and scientifically sound. The principal results are summarised in tables 1 and 2.

In table 1 the average transaction time includes machine processing time; the actual improvement in human data entry time averaged 30%. With regard to errors, in addition to the reduction in average error rate, they also report a reduction in the time spent correcting errors of 32%. Finally,

since transactions could be cancelled and started again at any point, cancel rates were a useful measure of how often users became totally confused; the comparison showed that with the modified programs 75% fewer transactions were cancelled.

Based on these data, Keister & Gallaway (1983) offer some projections showing the potential cost savings to customers at a typical application site with three data entry operators. In addition to the savings summarised in table 2, they suggest that improvements in user attitudes could result in less employee turnover and therefore in additional savings.

This example clearly demonstrates that attention to Human Factors in the software design of the human-computer interface can substantially improve the performance of the users and therefore the total system performance. As the authors conclude "this improvement is saleable to customers who are constantly looking for ways to reduce their operating costs".

Table 1 Performance with an improved interface design
(from Keister & Gallaway 1983)

Measure	Original	Modified	Improvement
Average transaction time (secs)	55	41	25%
Average errors in 60 transactions	35	26	25%

Table 2 Estimated cost savings to NCR customers
(from Keister & Gallaway 1983)

Type of savings	Amount of savings (in person-weeks per year)
Reduction of errors	1.6
Improved throughput	13.5
Reduced training (assuming 50% staff turnover rate)	9
Reduced relearning after forgetting	1

Combined per employee per year	17
Approximate dollar equivalent per year for typical customer site with 3 data entry operators	\$20k - \$30k



4.3 INTERFACE DESIGN - COSTLY KEYBOARD

This example is quoted, with his kind permission, from Chapanis (1986).

"Quite a different kind of example concerns the keyboard on IBM's PCjr. Late in 1983 IBM announced its newest entry into the small computer market with a great deal of fanfare because the PCjr was expected to set a new standard for the home computer market. But almost from the very beginning the PCjr was the object of widespread and vociferous complaints. Most of the complaints were directed at the computer's so-called 'chiclet' keyboard with its small keys that were at best awkward to use.

I am told by colleagues within IBM that the choice to go with this kind of keyboard was made by marketing against human factors recommendations. Whatever the true reason, it was a poor choice.

Suffice it to say, in July 1984 IBM announced that it was replacing the original keyboard with a full typewriter-style keyboard as standard equipment. Moreover, in a move that I believe is unprecedented in the computer industry and reminiscent of the many expensive automobile recall programs we have seen in recent years, IBM offered to exchange old keyboards for new ones, at no cost to current owners. We can only speculate on what that recall cost IBM. At the very least it must be in the millions of dollars, and that, by any standard, is an expensive object lesson about the costs of ignoring good Human Factors design."

4.4 WORKPLACE AND ENVIRONMENT

One of the best examples, proving the importance of ergonomics aspects of workplace and environment within the IT industry, comes from the 1984 'Coding War Games' reported by DeMarco & Lister (1985).

In this major exercise, 166 programmers (in 83 pairs) from 35 organisations took part; the two volunteer members of each pair worked in the same organisation at their own workplace, using the same support environment and the same language, and performed the same one day programming and implementation task working to the same specification. Careful time records were kept of their activities and interruptions etc. during the day, and full details were given of their working environment, followed finally by a questionnaire on the environmental factors. The work time to reach the performance milestones varied from best to worst by a factor of 5.6 to 1. Various checks were made and control studies were run to test the validity of the results, which were as follows.

In order to detect any correlation between environment and performance, the programmers were divided into four groups based upon performance; the average performance of those in the top 25% group was 2.6 times better than that of those in the bottom 25% group. The environmental factors recorded by the top group were then compared with those recorded by the bottom group, and significantly more of the top group had an acceptably quiet and private workspace, could silence and divert their telephone and were not often interrupted at their work. The environmental factor responses of the two groups are shown in Table 3.

The authors conclude that "changing the workplace from that typical of a lower-25% performer to that of an upper-25% performer offers the potential of a 2.6 to 1 improvement in the time to perform a complex programming activity".

Table 3 Environmental factor responses to questions by the top and bottom performing programmer groups (DeMarco & Lister 1985)

Environmental Factor	Top 25%	Bottom 25%
Dedicated floor space	78 sq ft	46 sq ft
Acceptably quiet workspace	57% yes	29% yes
Acceptably private workspace	72% yes	19% yes
Can you silence your phone?	52% yes	10% yes
Can you divert your calls?	76% yes	19% yes
Do people often interrupt you needlessly?	38% yes	76% yes
Does your workspace make you feel appreciated?	57% yes	29% yes
Ratio of performance	2.6 times better	

4.5 PRODUCT DEVELOPMENT

The development of the Portable Billing Machine (PBM) as part of the Plessey Immediate Billing System, provides an excellent case study of Human Factors applied throughout the development of an IT product (Davies 1981).

Although gas, electricity and similar bills can be processed by a computer, the procedure can take on average as long as five days from the meter-reader's first visit to the dispatch of the customer's bill. This time-lag occurs mainly because the procedure is organised via a central computer. This first generates slips for all the customers to be visited in each district; the slips are collated and distributed to district offices, the meters are read, and the completed slips are returned to the central computer for processing. Only then can bills be dispatched to customers.

An analysis of this system by Plessey Limited produced a scheme for streamlining these time-consuming procedures. Central to this scheme was the Portable Billing Machine, which would be carried by the meter-reader and used to deliver a bill directly to the customer at the time of the meter-reading. The specification for the PBM included a micro-processor with bubble memory, a visual display to show information from the micro-processor or as keyed in by the operator, a keyboard with numeric and function keys and a printer. In the new scheme the central computer still generates the customer lists, including such details as address, number of meters to be read, previous meter readings and tariff. These details are sent directly on-line to a local computer in each district office, which then produces the daily routes for each meter-reader and loads them, with the relevant information, into the individual PBMs ready for the next day's visits.

The meter-readers go on their rounds much as before, but now following the sequence of calls displayed on the PBM. Advice can readily be displayed about the exact location of the meters, any hazards to be avoided, etc. Where they can access the meter, they check and confirm the meter number and enter the meter reading. The software can check for any major errors, and at the press of a key can print out the bill. Where access to the meter cannot be obtained, an estimated bill can instead be printed and put through the customer's letter box.

Plessey recognised that human factors expertise would be vital for the project to succeed, since the new system must be not only efficient and timesaving but also acceptable and usable by the meter-readers. Ergonomics specialists were therefore brought into the project from the very start and played a formative role in defining important aspects of the system.

The design throughout was centred upon the ultimate users, the meter-readers. Since these potential users were the existing staff, and all of them were non-expert computer users who would have to use the machines on their own and far away from expert help, the design had to be as compatible as possible with the meter-reader's existing skills. Therefore it was decided that the system should step through the meter-reading procedure in a familiar way, requiring only a yes/no response from the operator wherever possible. For more detailed sub-routines to support supplementary decisions, function keys with suitable titles were chosen. The machine could thus be operated by a small set of simple rules which the meter-readers could learn quickly.

The ergonomists in the design team next advised on specific features of the interface and of the size, shape and weight of the unit itself. Aspects dealt with included the intensity of illumination needed for the visual display, to allow use in a wide range of lighting conditions; the design of the keyboard, to ensure best keying performance and minimum errors; and the design of the case and harness to make the PBM both portable for long periods and easy to operate in awkward or confined locations.

To evaluate the various proposed features, tests were run using actual meter-readers. In the laboratory first, tests were made on the treadmill to ensure that the PBM could be carried comfortably while walking for long periods, and also a dummy 'assault course' was constructed including a wide range of hazards and awkward situations described as typical by the meter-readers themselves, so as to test alternative methods for carrying and operating the PBM in awkward locations. Then tests moved into the field, and 8 of the 10 meter-readers in a selected pilot district went out on their rounds with prototype PBMs after being trained to use them. Both these sets of tests revealed improvements needed, which were then incorporated; the modified version was re-tested and the changes found to be successful.

The third and final phase of the Human Factors work involved the equally important aspect of training. This could not be as simple as might at first sight appear. First, trainees' skills would vary widely; some would be existing meter-readers, familiar at least with the existing procedures, but others would be new recruits unfamiliar with either procedures or computers. Second, the PBM had been designed to be flexible, to suit the needs of various billing organisations; some would use the built-in options and others would not, so that operational practices to be trained would vary for different clients. In addition, the training arrangements must allow for such practicalities as the variations in the number of trainees from 1 to about 50 in any one week, various recruitment patterns, availability of resources for training, and also the need for occasional training as existing readers leave and are replaced by unskilled recruits.

After considering many alternatives, the final scheme was chosen to allow maximum flexibility at economical cost. The scheme involved a self-contained training package which could be modified and finalised in close cooperation with the staff of the public utility concerned. The package consisted of slides, tapes, a manual, a trainer's handbook, a slide/cassette machine and a test PBM loaded with special programs and data. Trainees move through the programme listening to the training tape, pausing to look at slides, or to perform exercises when instructed. They can replay any section of the tape which they find difficult to understand. The programme is designed to introduce trainees to the PBM step by step, and to give them a clear picture of its operational features, so that they can appreciate how crucial a role they have as PBM operators in the whole system. Classroom and fieldwork training were combined as part of a complete one week regime with appropriate guidance and assessment by a supervisor.

This training programme itself was then tested using existing meter-readers

with varying experience from 0 to 16 years. Again, various improvements in the package were found to be needed and subsequently were included. However, both test groups used went through their field trials with ease - even the new recruit who had never been out on a round had no problems - thus demonstrating the ergonomic soundness of the training package and the ease of use of the PBM.

Comparative data are not available of the performance of the PBM against competitors, because these are US made and are largely imitators. However, the product certainly proved successful in the market place, and the Plessey Company was fully satisfied with the major human factors contributions throughout the whole process of product development. Finally, the product proved so successful that a separate company was established to supply portable billing systems; it is selling these products world-wide.



Figure 3. The Portable Billing Machine - developed with Human Factors from the very start - (a) the PBM (b) extensive tests with actual users were done in laboratory and field (c) tests on the treadmill for long term portability

4.6 SYSTEM INSTALLATION

The final example, which is perhaps most compelling in terms of cash savings, concerns human factors contributions to the organisation of the procedures and equipment involved in installing a large computer system. The example is adapted, with his permission, from an address by Chapanis (1986) to an Advanced Course on Human Factors in Informatics, sponsored by the British Science and Engineering Research Council and by the CEC CREST Committee. Chapanis writes as follows.

"Although they are somewhat different from the kinds of things that are usually associated with computers, problems of computer installation are of concern both to computer manufacturers and customers. And, it so happens, I have a good case study supporting the use of human factors in such an application. But first let me say a few words about why this is a matter of some importance.

The Installation Process

When a large computer or computer system has been manufactured, shipped out to a customer's shop, and delivered onto his loading platform, what happens? Obviously, it has to be installed. Boxes or crates have to be moved into the customer's facility; they have to be uncrated; the equipment has to be fitted into the space the customer has; it has to be assembled and attached to auxiliary systems, such as power and water if required; and finally the entire system must be connected together and tested as a whole.

In the case of large systems, installation is normally planned for long in advance. A year or more before actual delivery, the manufacturer's sales representatives, customer engineers, and systems engineers have been visiting the customer's shop, advising the customer and guiding him to a smooth and timely installation. How well all that planning works out in reality affects the manufacturer's costs and image, and the customer's productivity, profitability and satisfaction.

Look at it from the customer's standpoint. He's already been shown and told what the computer system can do. But when that equipment arrives on his platform, there are some additional human elements in the picture. Remember that the customer has had the equipment on order for several months, maybe even a year or more. Now he's got it, he's going to be using it, he's anxious to see it set up and running, and he may even be a little apprehensive about whether he will be able to use it properly and effectively. Moreover, getting the equipment set up and running always disrupts the customer's own activities to some extent. The sooner and the easier installers can get equipment working, the better.

The Background

The study I want to talk about shows how human factors improved the installation process for large computers resulting in substantial savings to the manufacturer, IBM in this case. The work was done by R E Granda and A Silvestro, two human factors specialists at IBM Poughkeepsie in 1977 and 1978. What led to the study were some serious concerns management had about the installability of the IBM system 370, model 168s and the 3000 series of computers. These concerns centred around the large amounts of time required to install the 168s, the large expenditures of manpower required for installation, and the much faster installation times achieved by some of IBM's competitors who had roughly comparable systems.

To take just one area of corporate concern, the greatest part of installability costs - up to 90% of them - are manpower costs. As you know, manpower costs are increasing, while hardware costs are decreasing. Because installation is such a labour-intensive activity, the more reductions that can be made in manpower costs, the greater the cost savings and other benefits to the manufacturer.

The Study Procedure

A task force was set up to investigate these problems and Granda and Silvestro worked closely with that task force. Although the task force studied many aspects of the installation process, I shall confine myself to only the human factors part of that work. First Granda and Silvestro studied current practices. From that study they defined and identified problem areas, after which they devised and made recommendations for improving the operation. Finally, the recommendations were tried out on a test installation of the 168 to check their feasibility before they were actually put into practice.

The Benefits from the Human Factors Improvements

Human factors improvements were made in ten major areas: Frame covers, Frame alignment, Console assembly, Cables, Pre-installation planning, Tail gate connections, Logic and installation manuals, Small parts, and Tool kit.

Without elaborating any more on the exact nature of the human factors improvements, what was the bottom line? It is shown in Table 3.

Table 3 Installation savings on IBM 3033s actually shipped to January 15, 1979, and extrapolated to December 31, 1979

Number of 3033s shipped	270
Total number of installation hours saved	40,770
Total savings due to reduction in installation time	\$1,630,800
Savings attributable to human factors changes	\$244,620
Net savings attributable to human factors changes (after deducting the costs of the study and of implementing the changes)	\$219,620
Net savings attributable to human factors changes (extrapolated to the end of 1979)	\$554,840

Installation savings were computed first for the 270 IBM 3033s that had been shipped to January 15, 1979. The net savings attributable to human factors changes, after deducting the cost of the study, amount to nearly a quarter of a million dollars. Extrapolating to installations that were planned for delivery up to the end of that year yielded net savings of over a half a million dollars. Actually, the savings are even greater than that because the lessons learned and the practices instituted because of this study have been applied to the installation of newer systems that are the successors to the 3033s. Taken together, these savings are significant even for a company as large as IBM!

To summarise, this is a success story relating to an often unrecognised aspect of large computer systems. A professional human factors team working in a field situation was able to make significant contributions to an important part of the company's business. These savings resulted in a reduced number of hours required for installation with a consequent savings in dollars. What these figures do not include is the increased productivity and satisfaction that customers must certainly feel because computers are now installed faster and with less disruption to their own business activities. Still another unmeasured ingredient is the increased satisfaction that customer engineers feel because of the reduction of the hard labour and physical tasks that they have to be involved in. Customer engineers now experience fewer frustrations and get home earlier. Though we cannot measure those things, they must surely be important and should not be ignored."

5. THE HUFIT PROGRAMME (HUMAN FACTORS FOR INFORMATION TECHNOLOGY)

5.1 Overview

This programme is intended to serve as a focus for Human Factors research activities within the scope of ESPRIT. The research is concerned with IT products from the moment of their conception, right through the design and development process, to their installation and use. Its particular contributions are aimed toward the development of an integrated human factors input to this whole process.

The programme consists of three project areas:

- A. From Conception to Use - an integrated Human Factors contribution to product design and development.
- B. Advanced Human-Computer Interfaces - theoretical and empirical investigations towards integrated user interfaces.
- C. Transfer of Human Factors knowledge - broad dissemination of Human Factors knowledge into the European IT community.

5.2 Objectives

HUFIT is ESPRIT's focal point for Ergonomics and Human Factors in the Office Systems programme.

The key objective is:-

1. to provide the European IT industry with a vital means of developing world class products.

These will be products, which

1. are more closely matched to the tasks, needs and characteristics of users,
2. are made feasible and economic by user-oriented design and development tools,
3. have effective, usable and flexible user interfaces.

5.3 Area A - From Conception to Use

The studies in this area will provide an integrated human factors contribution to product design and development.

Goals:

Project Area A aims to identify the critical steps in design that require Human Factors input and to incorporate human factors information into the IT product design process.

It will also develop tools which can deliver human factors information to designers in forms they can readily use during the product design process, from conception to installation and use. The outcome will be implemented in IT technology.

Activities:

- * Analysis of product design/development cycle
- * Collation and classification of Human Factors knowledge
- * Investigation of user needs and characteristics
- * Usability - its research, definition, design and evaluation
- * Development of usability design tools including an IT-based knowledge transfer mechanism

5.4 Area B - Advanced Human-Computer Interfaces

This area will comprise theoretical and empirical investigations towards integrated user interfaces.

Goals:

Project Area B aims to provide theoretical and empirical knowledge about basic interaction modes for application in products with integrated human-computer interfaces.

It will analyse and develop interfaces with improved usability and high efficiency. For future office tasks, more flexible interfaces will be needed which are adaptive or adaptable to user and task requirements. Area B will investigate and develop interfaces integrating different interaction modes.

Activities:

- * Development of theoretical and methodological knowledge for analysing cognitive tasks
- * Investigation of user performance, learning, skill acquisition and attitudes
- * Multimodal input/output techniques
- * Development of formal techniques for modelling and describing user interfaces
- * Theoretical and empirical analysis of dialogue structures and techniques
- * Prototyping and empirical evaluation of integrated, flexible interfaces

5.5 Area C - Transfer of Human Factors Knowledge

This area will involve the broad dissemination of human factors knowledge into the European IT Community.

Goals:

To ensure that the IT industry offers products which are usable and effective it is essential that human factors knowledge plays a part in their conception and development.

Project Area C is a programme to raise the level of awareness, knowledge and practice of Human Factors in the European IT industry.

Activities to be offered:

- * Seminars
- * Workshops
- * Consultancy Service
- * Assistance in setting up and developing new human factors centres in IT communities
- * Industrial affiliate programme

5.6 THE CONSORTIUM

The Faunhofer Institut fur Arbeitswirtschaft und Organisation (IAO), Stuttgart University FRG, accepted the main organisation of this work with the assistance of the Human Sciences & Advanced Technology (HUSAT) Research Centre, Loughborough University UK. These two organisations began the project in 1984 with the following companies as project partners: Bull (France), ICL (UK), Olivetti (Italy), Philips (The Netherlands) and Siemens (FRG). Also represented in the project as subcontractors to IAO and HUSAT are Munster University, West Germany; University College Cork, Eire; the Piraeus Graduate School of Industrial Studies, Greece and the University of Minho, Portugal. The scientific and technical content of the research programme is coordinated by IAO and HUSAT. The HUFIT project has been established to take on an integrating function for the area 'Human Factors and IT products' within the ESPRIT Programme; this means that the project should be closely linked with others and also that it should be opened to more partners at a later stage in its development.

5.7 SCOPE OF CURRENT WORK

The project work is progressing well towards all the goals and in all the activity areas listed above; the evaluation reviews have been very positive in approval and support of the programme. Some indication of the detailed work may be obtained from recent references - Bullinger et al 1987, Eason & Harker 1986, Eason et al 1987, Faehnrich et al 1987, Galer & Russell 1987, Hannigan & Herring 1987, Novara et al 1987 and Shackel 1986b.

Specific current work activities in Area A include:-

- analysis of the design and development processes undergone by IT products coming from large European manufacturers
- development of methods for describing task and user characteristics
- development of methods for implementing usability criteria and for evaluating models and prototypes during the design process
- gathering and classifying knowledge in computer human factors for use by the consortium and for dissemination to designers
- development of human factors tools to assist in IT product design

The work of Area B is concerned with the interaction of the user with the system. This involves theoretical and empirical analysis of advanced methods of human-computer interaction with the intention of creating prototypes of these interaction forms. Interaction forms under analysis are those consisting of the basic interaction techniques:- 'direct graphic manipulation', 'natural language' and 'formal language', and the work being done is as follows:

- formal modelling of human computer interaction
- operationalising the characteristics of different interaction techniques and the definition of generic interaction techniques
- developing tools for defining and implementing integrated multimodal user interfaces, as well as implementing pilot schemes
- establishing theory and practice in evaluation methods

In Area C the outcomes from the ongoing project are to be disseminated amongst interested European IT companies in the form of seminars for managers and designers' workshops, a consultancy service, an industrial affiliate programme and advice to companies on setting up human factors laboratories. As planned, these dissemination processes are being developed and piloted within the consortium during the current year 3 of the programme, and will be offered to the European IT industry from year 4.

Clearly it is not possible within the scope of this paper to mention in detail more than a small part of the wide range of work in hand within HUFIT. Therefore the following last three sections will summarise the work on three specific topics relevant to the three main programme areas.

5.8 BUILDING HUMAN FACTORS INTO THE DESIGN PROCESS

In order to develop suitable tools to deliver human factors knowledge and methods appropriately to designers, one must first have a clear picture of the design process. The generic seven stage model of product or system design, which is so widely taught and is a convenient framework for exposition, was not found to be a normal way of designing products. The findings agree with the views expressed by Whiteside (1986): "the coherent design process, into which information on user characteristics and requirements might be integrated, does not exist. The design process is disorderly and radically transformational, bearing little resemblance to the mythical state of affairs portrayed in the orderly stagewise diagrams found in textbooks and engineering manuals". Even those organisations that paid some attention to this form of process model appeared to use it only for reference points.

Eason and Harker (1986) report that there are many variations in the phases of the design process but it is usually possible to distinguish a general movement from specification through design and build to implementation. This, they argue, appears to be an 'invariant task structure' in that inevitably, to some degree, it is first necessary to establish the objectives of design, then to envisage and create a solution, and then to try it out.

The studies conducted as part of the HUFIT project (Olphert et al 1986, Hannigan & Herring 1986), based on the activities of the five supplier companies, have also indicated a generic design process. However, as Hannigan and Herring (1986) state, the coherent development process into which human factors tools can be introduced does not exist. The process is disorderly and represents a compromise between external factors such as new market requirements and internal factors such as availability of technology. The organisations, markets, and design structures vary widely across Europe. This finding concurs with the views of Dray and Monod (1986) who discuss this phenomenon and suggest that 'divergence' rather than 'convergence' is actually occurring. This indicates that it is even more essential to take account of the users' actual requirements in the development of the human factors tools. Although the content of the human factors input may be satisfactory in a variety of applications, the presentation style and medium may have to be tailored to each application within the design process.

The human factors tools being developed in this project take a number of forms but essentially are based around the identified requirements for design advice, task and user representation and usability assessment. The presentation modes include paper based materials such as guidance manuals and checklists, and computer based presentations such as CAI and the Decision Support System INTUIT.

The main design assumptions behind the evolution of an IT based human factors delivery system is that it should fit naturally within the integrated

environments visualised as being necessary for future fifth generation software development. INTUIT is intended to support the designer by use of the skills and experience of human factors experts in the development of the user interface. The long term aim of the INTUIT development programme is to amplify the design skills of the interface designer with the skills of the human factors specialist - who cannot be there in person - the paradigm being that of an expert partner.

5.9 MULTIMODAL HUMAN-COMPUTER INTERFACES

Human-Computer Interaction with separated modes like 'pure' Direct Manipulation or Natural Language has advantages and disadvantages. In conventional systems, these communication modes are usually realised separately. In many real situations, however, a combination of these communication modes would be considerably more adequate for the respective tasks or user groups.

Especially the combination of direct (graphical) manipulation and natural language interaction presents undeniable advantages (Hanne et al 1986). In many computer applications already working with graphic objects there is the easy possibility of combining natural language and graphic objects by pointing or selecting objects and operations. The expressive power of such combined natural language/direct manipulation interfaces in HCI is the possibility to allow deictic interaction on screen-oriented objects. The typical operations in this context are connected to words like 'this', 'that', 'here', 'there' and an accompanying selecting action. The idea of multimodal interaction consequently leads to the problem of deictic expressions and screen oriented pointing at 'objects of interest' which are anchored in the user's world.

This approach was the basis for the design and implementation of a direct (graphical) manipulation interface system (Hanne & Grable 1987). System-architecture and structure are based on layered models. The systems are developed and implemented on SUN workstations in 'C' and PROLOG based on UNIX, providing a set of modules and a communication layer for combined (multimodal) interfaces. The systems allow the inclusion of deictic/natural language references to objects represented on the screen. Several applications, a 'pure' direct manipulative interface to an expert system in the area of Aircraft Design, and systems allowing natural language/direct manipulation queries, have been developed.

5.10 CLASSIFICATION SYSTEMS

Another part of the HUFIT project involves two information support studies. The first is the GLOT (Glossary of Terms) exercise. This is the production of a multilingual glossary for the area 'Information and Communication Technology'. The glossary, with about 2000 terms and definitions has been submitted (Hoepelman et al 1986) and is currently being evaluated by external experts. Central to the project is a subsection involving the production of a glossary of software ergonomics.

The second study involves the collation and classification of computer human factors (CHF) knowledge. Already at HUSAT from 1982 onwards primary and secondary journals within the field of CHF, and materials from specialist conferences and from commercial abstracting and indexing services, are regularly searched and relevant material is added to the data base of bibliographic references. Access to the bibliographic data base is via a classification scheme and a thesaurus (Phillips 1985). The classification scheme and thesaurus have been developed using the user-centred design approach. The human factors experts in the consortium partners and elsewhere were the source of the classification scheme via workshops

(Phillips & Galer 1986). Pilot literature searches on Usability, Learnability, and Design Processes have been conducted for the partners to assess the usefulness and ease of use of the data base (Phillips 1986). Currently this is mainly a paper source; in the future the data base will be transferred onto a computer.

6. CONCLUSION

The aim of this paper has been to demonstrate to many managers and designers in the European IT industry, who are uncertain or unaware of the potential contribution from Human Factors, that there are substantial benefits to be gained. Indeed, I hope that the examples presented will have shown them that they cannot afford to ignore the knowledge and methods of Human Factors.

To point the way forward for those who wish to use Human Factors in future, the ESPRIT HUFIT project is now producing results and will in 1988 start its dissemination programme. In this first five years of its originally planned ten year programme, HUFIT has three main objectives: to discover and develop how to provide human factors knowledge and methods appropriately into the design process; to conduct research at the forefront of interaction structures and processes, so as to ensure for Europe the latest results relevant to advanced and integrated human-computer interfaces; to develop and establish a set of model approaches and techniques for disseminating Human Factors knowledge and methods most efficiently to managers and designers in the industry. I submit that the HUFIT Consortium is progressing well in meeting its deliverables and objectives. I believe that the HUFIT partners, and Human Factors specialists in Europe at large, are now ready to meet the demands which the IT industry ought to make upon them.

7. ACKNOWLEDGEMENTS

I wish to acknowledge the work of all the HUFIT Consortium partners, upon which I have drawn in preparing this paper. Especial thanks are due to senior colleagues at IAO and HUSAT who supplied specific material. However, nothing here should be taken to commit any partner, and the opinions and any errors are my own.

8. REFERENCES

- Bennett J L (1979) The commercial impact of usability in interactive systems. In Shackel B (Ed) Man-Computer Communication, Infotech State-of-the-Art Vol 2; Maidenhead UK, Infotech International.
- Bewley W L, Roberts T L, Schroit D & Verplank W L (1983) Human Factors testing in the design of the Xerox's 8010 'Star' Office workstation. Proc. Conf. CHI'83 Human Factors in Computer Systems, pp 72-77. ISBN 0-89791-121-0. ACM, PO Box 64145, Baltimore, MD 21264.
- Boies S J, Gould J D, Levy S, Richards J T & Schoonard J W (1985) The 1984 Olympic Message System - a case study in system design. IBM Research Report RC-11138. IBM T J Watson Research Center, PO Box 218, Yorktown Heights, NY 10598.
- Bowman B L (1986) Cross-functional collaboration: teaming for technological change. In Brown O & Hendrick H W (eds) 1986 pp 511-515.
- Branscomb L M (1983) The computer's debt to science. Perspectives in Computing, 3.3, 4-19 (published by IBM).
- Brown O & Hendrick H W (1986) (Eds) Human Factors in Organisational Design and Management II. Amsterdam, North-Holland.

- Bullinger H-J, Faenhrich K-P & Ziegler J (1987) Software Ergonomics - present state and developmental trends. Proc. Internat. Conf. on Human-Computer Interaction, Hawaii; to be published.
- Burch J L (1984) Computers: The Non-Technological (Human) Factors. Lawrence KA, The Report Store. ISBN 0-916313-00-X.
- Cakir A, Hart D J & Stewart T F M (1980) Visual Display Terminals. Chichester, Wiley. ISBN 0-471-27793-2.
- Carpentier M (1984) Opening Address to the First ESPRIT Technical Week. In Roukens J & Rennart J F (Eds) ESPRIT '84 - Status Report of Ongoing Work, pp xiii - xx; Amsterdam, North-Holland 1985; ISBN 0-444-87740-1.
- Chapanis A (1985) Training and civilizing computers. Annals of the New York Academy of Sciences, 426, 202-219.
- Chapanis A (1986) The business case for Human Factors in Informatics. Proc. SERC-CREST Course on Human Factors for Informatics Usability; HUSAT Research Centre, Loughborough University, to be published.
- Damodaran L, Simpson A and Wilson P (1980) Designing Systems for People. Manchester, NCC Publications. ISBN 0-85012-242-2.
- Davies D G (1981) Ergonomics in the development of the Portable Billing Machine. Pergamon-Infotech State of the Art Report on User Friendly Systems; Maidenhead UK, Pergamon-Infotech. Adapted and illustrated in National Electronics Council (1983).
- DeMarco T & Lister T (1985) Programmer performance and the effects of the workplace. Proc. 8th International Conference on Software Engineering, 28-30 August, pp 268-272. ISBN 0-8186-8620-0. Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854.
- Douglas D M & Marquis F A (1985) Supporting computer user staff in an automated office - a case study assessment. In Brown I D, Goldsmith R, Coombes K & Sinclair M A (Eds) Ergonomics International'85, pp 325-327. London, Taylor & Francis.
- Dray S M & Monod E (1986) The new 'internationalism' in macro ergonomics. Proc. Second International Symposium on Human Factors in Organisation Design and Management; Amsterdam, North-Holland.
- Eason K D (1982) The process of introducing information technology. Behaviour and Information Technology, 1, 197-213.
- Eason K D (1984) Towards the Experimental Study of Usability, Behaviour and Information Technology, 3.2, 133-143.
- Eason K D & Harker S D P (1986) Effective Human Factors Contributions to the Design Process. Proceedings of the SERC CREST course Human Factors for Informatics Usability; HUSAT Research Centre, Loughborough University.
- Eason K D, Olphert C W, Novara F, Bertaggia N & Allamanno N (1987) The design of usable IT products - the ESPRIT/HUFIT approach. Proc. Internat. Conf. on Human-Computer Interaction, Hawaii; to be published.
- Faenhrich K-P, Davies D G & Ziegler J (1987) HUFIT - Human Factors in Information Technology. Proc. Internat. Conf. on Human-Computer Interaction, Hawaii; to be published.

- Gaines B R (1984) From Ergonomics to the Fifth Generation : 30 Years of Human-Computer Interaction Studies. Proc. INTERACT'84 IFIP Internat. Conf. on Human-Computer Interaction pp 1-5; Amsterdam, North-Holland.
- Galer I A R & Yap B L (1980) Ergonomics in intensive care: applying human factors data to the design and evaluation of patient monitoring systems. Ergonomics, 23.8, 763-779.
- Galer M & Russell A J (1987) The presentation of Human Factors to designers of IT Products. Proc. INTERACT '87 IFIP Internat. Conf. on Human-Computer Interaction; Amsterdam, North-Holland.
- Grandjean E & Vigliani A (1980) Ergonomics Aspects of Visual Display Terminals. London, Taylor & Francis.
- Grandjean E (Ed) (1984) Ergonomic and Health Aspects in Modern Offices. London, Taylor & Francis.
- Hanne K H, Hoepelman J Ph & Faehrich K-P (1986) Combined Graphics/Natural Language Interfaces to knowledge-based systems. Proc. Conf. Artificial Intelligence and Advanced Computer Technology; Liphook UK, TCM.
- Hanne K H & Grable A (1987) Design and implementation of direct manipulative and deictic user interfaces to knowledge based systems. Proc. INTERACT '87 IFIP Internat. Conf. on Human-Computer Interaction; Amsterdam, North-Holland.
- Hannigan S & Herring V (1986) The Role of Human Factors in the Design of IT Products. Deliverable A1 2b ESPRIT project no 385.
- Hannigan S & Herring V (1987) Human Factors in office product design - European practice. Proc. Internat. Conf. on Human-Computer Interaction, Hawaii; to be published.
- Hendrick H W & Brown O (Eds) (1984) Human Factors in Organisational Design and Management. Amsterdam, North-Holland.
- Hoepelman J, Heller N & Thiele S (1986) Revised Draft of Glossary of Terms: Working Paper C6.3, HUFIT Code: HUFIT/9-IAO-6/86.
- HUSAT (1983) Social Security Human Factors Strategy. HUSAT Research Centre, University of Technology, Loughborough, U.K.
- Keister R S & Gallaway G R (1983) Making software user friendly: an assessment of data entry performance. Proc. Human Factors Society 27th Annual Meeting, pp 1031-1034; Santa Monica CA, The Human Factors Society.
- Marquis F A & Douglas D M (1985) User acceptability of a computer system - a case study. In Brown I D, Goldsmith R, Coombes K & Sinclair M A (Eds) Ergonomics International'85, pp 76-78. London, Taylor & Francis.
- Martin J (1973) Design of Man-Computer Dialogues. Englewood Cliffs NJ, Prentice-Hall.
- Moore T G & Dartnall A (1982) Human factors of a microelectronic product: the central heating timer/programmer. Applied Ergonomics, 13.1, 15-23.
- National Electronics Council (1983) Human Factors in Information Technology. Published by the National Electronics Council, UK; ISBN 0-9508590-0-1, distributed by Wiley, Chichester, UK.

- Nickerson R S (1969) Man-computer interaction: a challenge for human factors research. *Ergonomics*, 12.4, 501-517.
- Novara F, Bertaggia N, Allamanno N, Fox S & Olphert W (1987) Usability evaluation and feedback to designers - an experimental study. Proc. INTERACT '87 IFIP Internat. Conf. on Human-Computer Interaction; Amsterdam, North-Holland.
- Olphert C W, Galer M D, Hannigan S & Russell A J (1986) Design Cycle Model. Deliverable A1 1b ESPRIT project no 385.
- Phillips K E (1985) Classification of Domains of Human Factors in Information Technology; Working Paper A3.1a, HUFIT Code: HUFIT/1-HUS-5/85.
- Phillips K E (1986) A Pilot Search on the Computer Human Factors Data Base - Topic: Usability; HUFIT Code: HUFIT/5-HUS-7/86.
- Phillips K E and Galer M D (1986) The Development of a Computer Human Factors Classification and Collation of Human Factors Knowledge; Interim Report A3.1, A3.2; HUFIT Code: HUFIT/4-HUS-8/86.
- Sackman H (1970) Man-Computer Problem Solving. Princeton, Auerbach.
- Shackel B (1959) Ergonomics for a computer. *Design*, 120, 36-39.
- Shackel B (1969) Man-computer interaction: the contribution of the human sciences. *Ergonomics*, 12, 485-499.
- Shackel B (1985a) Ergonomics In Information Technology In Europe - A Review. (Based upon a report for the Commission of the European Communities in 1984) *Behaviour & Information Technology*, 4.4, 263-287.
- Shackel B (ed) (1985b) Human-Computer Interaction - INTERACT '84. Proc. IFIP International Conference; Amsterdam, North-Holland.
- Shackel B (1986a) IBM Makes Usability as Important as Functionality. *The Computer Journal*, 29, 475-476.
- Shackel B (1986b) Usability - context, framework, definition, design and evaluation. Proceedings of the SERC CREST course Human Factors for Informatics Usability; HUSAT Research Centre, Loughborough University.
- Shackel B & Eason K D (1986) Organisational prototyping - a case study in matching the computer system to the organisation. Paper to Second International Symposium on Human Factors in Organisational Design and Management, Vancouver, 19-21 August; to be published.
- Smith H T and Green T R (eds) (1980) *Human Interaction with Computers*. London, Academic Press.
- Weinberg G M (1971) *The Psychology of Computer Programming*. New York, Van Nostrand Reinhold.
- Whiteside J A (1986) Classifying Users: A hard look at some controversial issues. Panel in CHI 86 Conference Proceedings; New York, Association for Computing Machinery.
- Winograd T (1972) *Understanding Natural Language*. Edinburgh, The University Press.

Basic Concepts of the Functional Analysis of Office Requirements

Gunter Schäfer BIFOA, Cologne
Rudy Hirschheim Templeton College, Oxford

The Functional Analysis of Office Requirements (FAOR) project - ESPRIT Project #56 which completed in May 1987 - has produced a comprehensive approach (methodology) for office requirements analysis. The purpose of FAOR is to define the scope and objectives of an office analysis, and to tailor an appropriate investigation method and set of instruments for undertaking the analysis. The method and instruments may then be applied to the specific organizational situation for which they were tailored. A key feature of the approach is the incorporation of benefits analysis techniques which are employed to determine the benefits and detractions of applying various office system scenarios. The result should enable an organization to choose the office system which best supports the performance of its business objectives.

1. Introduction

Because every office workplace is potentially affected by the introduction of new office systems, implementing these systems means large investments in an area where previously there have been few. Since a large share of the costs of running an organization stems from office work, any improvements in productivity promises considerable economic return. Improvements in office administration may also be translatable into competitive advantage.

Experience shows, however, that benefits do not come about automatically through the application of information technology (Hirschheim and Feeny 1986). They are achievable only if the technology is designed and utilized according to the specific needs of the organization - in other words, if it is carefully tailored or tailor made. The successful use of information technology requires an analysis approach which takes into account a thorough understanding of the organizational role of office work and its consequences for the application of office technology. More specifically:

1. Office work is neither an independent function of an organization nor its sole task. It can only be understood within the context of its interconnections with other areas of work, such as production and sales.
2. Office work needs to be supported by office technology rather than automated. The technology provides tools which the office worker can use to perform his tasks.
3. Office technology cannot be seen as the sum of isolated applications or as affecting only certain workplaces. Although considered a concept of its own, office technology only successfully functions if it interfaces closely with other kinds of information technology applications.

It is therefore necessary to view office technology in a broader organizational context. Office technology and its applications need to be seen in terms of an 'office system' - an amalgam of technologies, people, procedures, and applications which support the office worker in carrying out organizational duties. These duties are typically conceived in terms of office activities and tasks which are undertaken to achieve the objectives of the organization.

In order to develop an appropriate and effective office system, a suitable analysis approach or methodology is needed. The FAOR approach (Functional Analysis of Office Requirements) aims at providing an analysis method for the office domain (Schäfer 1987b). It focuses on the early stages of the life cycle of an office system where the determinants of a successful application of office technology have to be identified and, based on operational objectives, have to be translated into requirements for the application. The requirements reflect what the office system is supposed to do and specify its features sufficiently for the subsequent design.

2. Objective of FAOR

The tools provided by office technology have often been seen as supporting isolated office activities. In this case, office systems design for a particular organization is merely a selection from a range of available and general office tools. This approach, however, neglects the differences existing in organizations. The nature of office work as supporting the achievement of organizational functions, demands that these functions and not the office activities be at the center of attention when support requirements are being identified.

Furthermore, introducing an office system is not purely and not even mainly a technical issue. Its widespread implications on the way office work is carried out make it to the same extent an organizational and social issue as well. Offices should therefore be considered socio-technical systems. An approach for analyzing offices has to perceive these factors and their interrelationships and to identify requirements suitable for the organization as a whole and not merely its technical aspects.

This requires an approach which is flexible enough to take the individual situation of the organization into account; that is, it has to contain powerful tools for understanding the specific 'problem situation' in its entirety and to adjust its investigation and analysis as close as possible to this individual situation.

3. Principles of FAOR

The considerations outlined above have led to a number of principles for the FAOR approach. They reflect its basic understanding of the office domain and how analysis of the office should be conducted. These principles are the key for understanding the structure and inherent logic of the FAOR approach.

Principle 1: Comprehensive Approach to Office Analysis

In order for any approach (methodology) to be truly effective for office requirements analysis, it must be *comprehensive*. It must permit the analyst to build models of the office domain (by providing suitable 'framing mechanisms' or 'model building tools'), and it must contain appropriate 'principles of operation' which permit the analyst to undertake the necessary analysis activities in a particular office context (Welke 1983).

Although there exists a number of approaches or methodologies for office systems requirements analysis, all appear deficient in one sense or another. Some focus on 'model building', relegating 'guiding principles' or 'principles of operation' to secondary or tertiary concern. Others are the reverse (Hirschheim 1985). The FAOR approach seeks to overcome both biases, offering a vehicle for model building while providing principles of operation. FAOR is unique in this respect.

Principle 2: Multiperspective Analysis

Analysis is never value-free nor for that matter is office work. Both the nature of the problems the organization is facing and the question of 'what an office actually constitutes' and 'what purpose it serves' is subject to a number of very different interpretations. This is particularly pertinent

in the analysis of human action and the specification of a technical system whose success is largely dependent on the human.

Analysis is typically guided by one *perspective*; each analyst adopts a particular perspective. Without a deliberate *multiperspective* scope, the analyst as the observer of reality, interprets what he sees only through one preexisting subjective filter. The value judgement implied ascribes to the various aspects of reality he perceives a certain importance and thus determines where and how office systems may be applied to support office work. Only by recognizing that there exists more than one way of viewing the world can the analyst understand the office and potential problems in connection with office work. He must accept different interpretations as being equally valid. There is a growing awareness of the need for adopting a multiperspective approach (cf. Wood-Harper et al. 1985, Bjerknes and Bratteteig 1985, Lanzara and Mathiassen 1985)

Principle 3: Functionality of Requirements

Applying different perspectives is essential for understanding the problem situation well. When it comes to specifying requirements, the ambiguity of coexisting interpretations due to different perspectives and the various facets of the problem situation has to be resolved. The vehicle for dealing with this in FAOR is through the focusing on the *functionality of requirements*. This means, FAOR forms the basis for subsequent design of an office system which supports the client organization in the performance of its functions and thus, supports the achievement of the objectives of the organization.

By using the concept of *function*, FAOR assumes the existence of an organizational 'interest' distinct from the interest of the people involved or as some common denominator everybody accepts as valid beside his own interests. This perception, however, leaves FAOR with the task of dealing with the natural fuzziness of what constitutes the objectives of an organization and how they can be investigated, specified and evaluated.

An analysis which leads to *functional requirements* is more than an analysis of functions. Although perceiving support requirements needs an understanding of what an organization is supposed to do, it is also an investigation into the functionality of how this can be achieved and the role an office system may play (Krückeberg 1983). Distinguishing between the terms 'functional' (or 'functionality') and 'function' is important for FAOR. 'Function' is an abstraction of the main activities carried out to achieve organizational objectives (e.g. a marketing function). 'Functionality' (of requirements) qualifies the way activities are supported by an office system.

Principle 4: Method Tailoring

If the problem situations and the organizational objectives a method is designed to deal with can vary substantially, it does not make sense to use tools and guidelines in the same way in all cases. Instead a high degree of flexibility of a method is desired which allows the method to be adjusted to the individual client situation. Traditional methods incorporate a number of parameters which allow an adjustment to factors, such as the level of detail or the scope of the study.

FAOR goes a step further. Its investigation and analysis method is individually constructed to suit the analysis situation in an optimal way. This 'method tailoring' is done on the basis of a broad and open exploration of the office under investigation. Method tailoring, however, does not mean that every time the FAOR approach is applied a method is build totally from scratch. FAOR uses the concept of a *tool box of prepackaged instruments* which are selected, combined and adjusted to fit the client situation. A better fitting method promises advantages both in terms of method economy and the quality of results.

Principle 5: Framework of Thinking

Method tailoring implies that instruments are used as some kind of reference on how to conduct an analysis and investigation, rather than as modules which can be flexibly combined. This means that in method tailoring the tools and techniques of the various instruments can be drawn upon freely and a new sequence of steps can be constructed. The analogy to an orchestra may be made where musical instruments of very different nature are able to play a wide variety of melodies. FAOR prefers coordinated action using different means to the other option of a predefined sequence of steps.

In order to do this in an operational way applicable in practical settings, a form of guidelines must be employed which are non-procedural. While traditional methods incorporate the knowledge about the office domain and the practical investigation and analysis guidelines, FAOR references this knowledge when needed. FAOR bases its approach not only its method-tailoring part on a general problem solving and learning heuristic, the so-called *Soft Systems Methodology* (SSM) developed by Peter Checkland (1981). It is a loosely coupled framework of systems thinking concepts which satisfies the FAOR requirements of being very open in terms of the issues to be tackled as well as the utilization of supplementary knowledge.

The instruments are not the only elements referenced in FAOR. It also incorporates reference knowledge on the structure and issues of the office domain in the *Generic Office Frame of Reference* which contains predefined perspectives of the office. Furthermore, the *Benefit Analysis Framework* contains reference knowledge about the process and criteria of evaluation of office systems. Figure 1 shows the relationships between the FAOR elements.

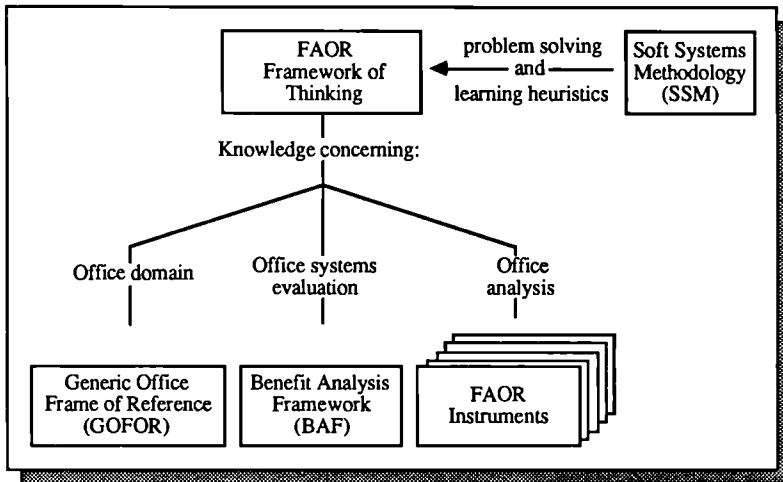


Figure 1: Use of the FAOR elements in a framework of thinking

Principle 6: Recursive and Participative Investigation

The flexibility of the Soft Systems Methodology is used for achieving another major principle. FAOR aims at a stepwise delineation and specification of an office system, by defining requirements. The understanding of both the office situation and the office system is successively refined and detailed in the course of a study.

In the early phase of a FAOR investigation, the major emphasis is given to the issue of focusing on the relevant problems. This knowledge is translated into a *model* (or a number of alternatives) of an office system capable of solving the identified problems. It is then successively firmed up in a

stepwise approach, always with the whole system in focus, and not only its technical aspects. The model of the office system is developed based on a very broad and open exploration in order not to get fixed prematurely on a certain technology which may assist only in overcoming secondary problems. Moreover, the model of the office system developed this early, describes its organizational role and is not technology oriented. Incorporated in SSM is the notion that obtaining information on the problem situation is done in interaction with the members of the organization. The importance of their involvement has been emphasized by Mumford (1983) and Pava (1983). The model of the office system is the basis for repeated discussions with the client and other members of the organization. These discussions are in the earlier phases of FAOR the major input for questioning the current understanding and for deepening it. Only after the nature of the office system and its major tasks have been clarified and agreed upon can the instruments be applied to provide more detailed information.

Principle 7: Benefit Evaluation

Since a model of an office system which is elaborated very early can only be tentative in nature, it is important for FAOR to continuously question it for its benefits to the organization and to its users. In FAOR terms, it is the functionality aspect which must be satisfied by the concept of an office system.

Because this is essentially an evaluation process, a part of the reference knowledge of FAOR concerns *criteria and issues of evaluation*. They are contained in the Benefit Analysis Framework and applied to the concept of an office system at various stages of its elaboration and not only as a post-design or post-implementation exercise. As such, the results of early evaluations, although they are not totally accurate, should contribute to the understanding of support requirements in the office (Schäfer 1987a).

4. Frames and Analysis of Office Reality

As already pointed out, FAOR contains two kinds of knowledge about the office. While the Generic Office Frame of Reference (GOFOR) specifies the aspects which make up the office, a number of instruments provide the practical means for office analysis.

4.1 Generic Office Frame of Reference

GOFOR employs the concept of perspective for specifying the relevant aspects of the office domain. The perspectives are means for coping with the complexity of offices by separating domains of office functioning, one at a time, and for bringing them together as building blocks for constructing a model of the given office. The way aspects of the office are divided into perspectives can be understood through a visual analogy where people have a certain viewpoint in looking at how an office functions. Because the perspective corresponds to a certain viewpoint, the aspects of office functioning stressed or prioritized come to the foreground.

For example, a manager may have the responsibility of getting things done and therefore takes a viewpoint which emphasizes the tasks to be carried out. Other aspects, such as resources, then take a subsidiary role. This example also shows that the office aspects considered in one perspective are only in the foreground and not independent from those of other perspectives. A person, like the manager, will not see the office only from one perspective, according to his immediate task he will shift his viewpoint in order to accommodate other aspects. As required, the office manager will also consider the resources necessary for carrying out the tasks and thus link two perspectives.

GOFOR describes seven reference perspectives which are of particular importance in the functional analysis of offices.

1. The *function perspective* addressing aspects of intention and purposiveness of office work.

Three closely interrelated implementation perspectives stressing issues and aspects of the performance of office work:

2. *Information processing perspective*
3. *Resource perspective*
4. *Personnel perspective*

Three management perspectives:

5. *Coordination perspective*
6. *Control perspective*
7. *(Re-)Organization perspective*

GOFOR characterizes the perspectives by naming the aspects contained in a perspective and their interrelationship and by showing how offices can be modelled within a perspective using *Petri nets* (Petri 1980).

4.2 Instruments

The FAOR instruments provide the practical means of using aspects of these perspectives as part of a multiperspective office analysis. Instruments are used within each FAOR application to support the analyst in gathering information about relevant aspects of the office and organizational situation, which is then used for the generation of models of the situation using appropriate modelling techniques, as well as to focus the analysis subsequently so as to generate a sufficient specification of requirements.

Each FAOR instrument is a prepackaged set of *tools, techniques and application principles* which is tailored to accomplish a particular investigation, analysis, modelling or evaluation task from one or several perspectives. Not unlike a tool box, instruments provide the analyst with a range of method building blocks which are selected, tailored and combined prior to application, according to the aspects and objectives of a given client study (Oppelland *et al.* 1977).

The *Function Analysis Instrument (FAI)* has the task of understanding a given office as a functional system within the organization, i.e. to identify, define and characterize the essential functions the office serves in order to fulfill the organizational objectives and to determine complementary and contradictory interrelationships between functions. This allows the purpose of the planned office system to be perceived in terms of the organizational functions and to specify clearly what should be achieved by implementing the office system.

The *Communication Analysis Instrument (CAI)* perceives the office as a communication system in which both individual office personnel and functionally related work groups transfer information objects primarily for the purpose of fulfilling organizational objectives. A number of GOFOR perspectives are applied to focus the analysis in turn on the existing pattern of office communication, on the flow of information objects, on the task, role and personnel aspects which influence the context in which communication occurs and on existing and planned resources (particularly communications technology) which will constrain any opportunities for change.

The *Information Analysis Instrument (IAI)* concentrates on information objects. These may be documents on multiple media, simple messages or even spoken phrases. These information objects exhibit many characteristics an office system must be able to cope with, not only in terms of indi-

vidual information objects, but also in the ways they can be interrelated, e.g. the existence of versions or copies of the same document.

The *User Needs Analysis Instrument (NAI)* focuses on the human being as the user of the office system which is going to be designed. His interests and preferences are valuable information for identifying requirements since the success of an office system to a large extent depends on how well the office worker accepts the change and on the way he will make use of the office system according to his own working style.

Many current office analysis methods can be compared directly with one or another FAOR instrument. Often they are biased by a particular perspective on the office without making this perspective explicit. The FAOR approach remains open to the incorporation of further 'external' instruments, as long as the perspective(s) they take can be made explicit.

5. Office Systems and Evaluation

A similar relationship as between the GOFOR and the four office analysis instruments exists between the Benefit Analysis Framework (BAF) and the *Benefit Analysis Instrument (BAI)*. Figure 2 shows the relationships between the FAOR frameworks and instruments.

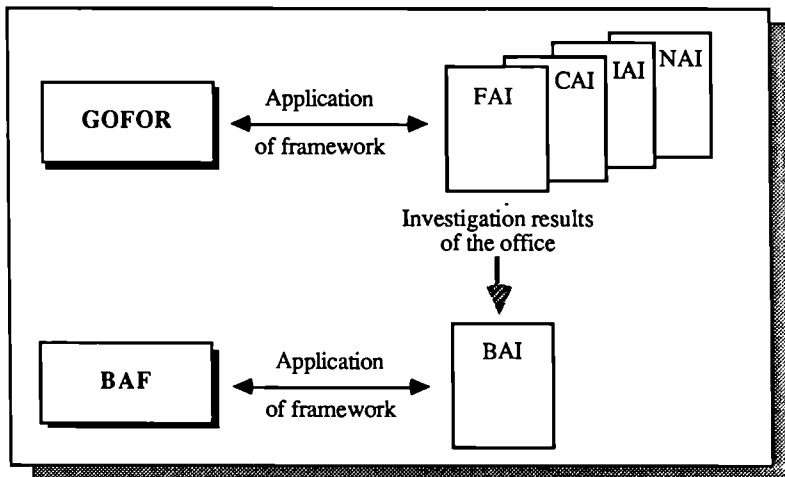


Figure 2: FAOR Frameworks and Instruments

The BAF contains reference knowledge about *principles of office systems evaluation*. It starts with the difficulties of evaluating the impact of a technology where only a few benefits can be captured on a quantitative, monetary basis. This is more or less a general phenomenon of information technology based systems. The analyst increasingly realizes that such evaluations are inherently subjective, qualitative and not very meaningful on the basis of formal prescriptions (Hirschheim and Smithson 1987, Legge 1983). This situation is aggravated in a benefit analysis performed before the office system is actually implemented because effects have to be predicted rather than observed, often only on a sketchy model of the future system.

In order to cope with this situation, BAF attempts to provide concepts, criteria and a sequence of fundamental evaluation steps which help to make the evaluation process transparent. To state clearly the assumptions, the line of argumentation, and the evaluation criteria which are applied is as important as the results. With the fundamental steps of evaluation, BAF attempts to guide the

analyst's thinking in terms of cause-effect chains which are naturally involved if he interprets changes related to an office system in terms of benefits for the organization.

The BAF is focused on the office and office systems. Benefit concepts, the notions of efficiency, effectiveness and needs satisfaction have been selected in order to cope also with the important impact of office systems on the quality of work results and on social aspects.

The BAI shows how this framework is applied in a FAOR application. There may be very different purposes for a benefit analysis according to the development stage of a systems specification. A benefit analysis dealing with a rough office system model faces different conditions from an evaluation with an elaborated requirements specification, or one which is carried out after the office system has been implemented. The BAI outlines the implications of the different situations on the conduct of the benefit analysis. The actual information gathering, however, is mainly carried out using the other four instruments.

6. Office Analysis and FAOR Activities

The FAOR approach includes an *Activity Framework* which structures the sequence of events which takes place in office analyses. The activity framework embodies four general activities where the activities specify the tasks to be carried out and define intermediate baselines in an office systems life cycle (figure 3).

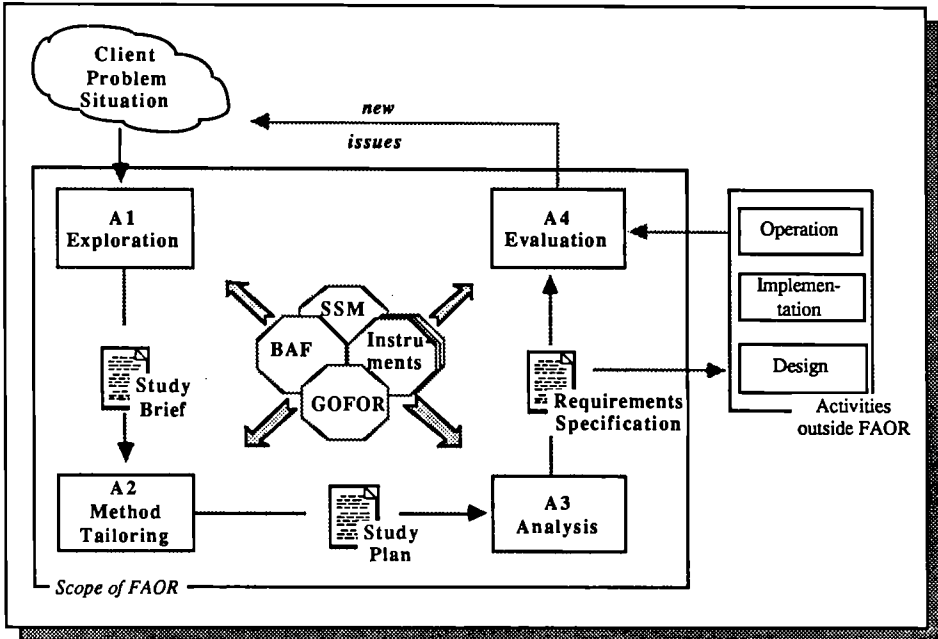


Figure 3: The scope of FAOR

The activities are bound together on the basis of content or underlying intention. They are deliberately meant not to form a strict sequence of phases. Although baselines are placed to denote intermediate results, going from one activity to the next does not imply a strict termination of this activity. The analyst can always come back to further elaborate on a previous activity if he feels the need to do so. Each activity has a certain task to do in which it can draw upon GOFOR, BAF and the instruments for assistance.

Instead of a strict sequence of guidelines, the activities are carried out based on the major notions of *Soft Systems Methodology (SSM)*. Its emphasis on 'soft' or ill-defined and ill-structured problems suits the FAOR understanding of the socio-technical nature of offices and the importance of organizational as well as social issues in the design of office systems. Furthermore, SSM is a *systems thinking methodology* and not limited to a certain subject area. Since the specific problems faced in the various FAOR activities are of this soft nature, SSM qualifies as a learning and problem solving approach which can underlie each FAOR activity.

SSM makes operational 'soft systems thinking' through the concept of 'human activity system' which

can manifest itself only as perception of human actors who are free to attribute meaning to what they perceive. There will thus never be a single (testable) account of a human activity system only a set of possible accounts all valid according to particular viewpoints or *Weltanschauungen*. (Checkland 1981)

The problem solving cycle of SSM builds on concepts for a broad investigation of the problem situation as unbiased as possible and for conceptualizing the problem situations in terms of systems. The conceptual interpretation of the problem situation as a system starts in SSM from one or a number of hypotheses about the purpose underlying the human actions which can be observed. Therefore, human activity systems are seen as purposive systems. The systems concepts are used to better understand the reality, and to see where changes may be applied to improve the reality.

There are four activities in FAOR: *Exploration (A1)*, *Method Tailoring (A2)*, *Analysis (A3)* and *Evaluation (A4)*. They are perceived as constituting problems of their own, and can be solved by applying SSM for each activity but with different tasks.

In the exploration activity (A1), SSM assists an analyst in getting to know the client organization and in developing a rough model of the future office system. In the method tailoring activity (A2), SSM is primarily a thinking tool for constructing a method out of the existing instruments according to the study objectives defined in the exploration activity. In the analysis activity (A3), SSM helps to relate the results of applying individual instruments to a coherent set of requirements. In the evaluation activity (A4), it is used to control the study results towards the fulfillment of the study objectives.

Activity A1: Exploration

The purpose of A1 is to obtain a broad, rather than an in-detail *understanding* of the context and background of problems the client organization is facing. Aspects which the analyst thinks relevant will be explored further and the findings are condensed into one or more so-called 'rich pictures' - a particular SSM technique which depicts complex problems in a very compact way. This helps the analyst to structure the problem situation conceptually and to analyze it for consistency and completeness. In particular, the potential of rich pictures as a tool for communication with the client, or even the possibility of elaborating the rich picture together with an interviewee, greatly assists the analyst in what typically is an unstructured activity of information collection and comprehension.

On this basis, one or more 'relevant systems' are outlined. They have the status of an hypothesis of the eventual improvement of the problem situation by means of changes which seem likely to be both 'feasible and desirable'. Thus, the relevant system defined in A1 outlines a preliminary and rough model of how support by an office system can be achieved. This model, constructed as a 'notional system', is the basis for discussing the study scope with the client. This will also reveal constraints both for the study, e.g. resource limitations, and for the requirements, e.g. the need to take into account the equipment of a certain manufacturer. Of particular importance are the study objectives. They determine the desired results of the study, e.g. the level of requirement detail and the comprehensiveness of the requirements specification. The results of A1 are summarized in a study brief. It documents the common understanding of the analyst and the client of the problem situation and how this should be remedied.

Activity A2: Method Tailoring

In method tailoring, the analyst has to *determine the means* for solving the issues addressed in the study brief. The analyst has to decide what he needs to know about the organization and with what tools and techniques for investigation, modelling and analysis he can obtain this knowledge. At the same time, he must overcome the constraints of the situation.

Method tailoring is essentially a comparison between analysis problems identified in A1 and analysis capabilities available through the instruments. Non-FAOR instruments (usually termed methods themselves) may also be considered in this process. Available instruments constitute possible 'pre-structurings' of the analysis covering different aspects. They correspond to potential relevant systems in the SSM sense. Thus a relevant system in the method tailoring activity indicates that the analyst sees the problem situation to be best treated by a given instrument.

Defining relevant systems to fit the problem situation does not mean that they should cover the whole range of the problem situation. Rather the task of method tailoring itself suggests that most problem situations cannot be satisfactorily dealt with by only one instrument. Thus, the relevant system is likely to cover only a part of the entire problem situation. A number of relevant systems therefore have to be chosen to cover the range of problems and supplement each other.

Although in the first instance, this comparison will be carried out with each instrument individually, the real problem is their combination in a method which is both effective and economic. The analyst has to see which combination of instruments he should use and how he should combine and adjust them to form one coherent method.

The results of A2 are summarized in a study plan. It outlines the tools and techniques to be used in requirements analysis, as well as the overall structure and organization of the investigation project.

Activity A3: Requirements Analysis

In activity A3, the method tailored in A2 is applied. The analyst tries to gather the information he needs about the client organization by applying the tools and techniques specified in the study plan. The primary objective, however, is the *identification of the requirements*. The practical guidance of how to use the tools and techniques and which aspects to emphasize in the modelling and analysis is supplied by GOFOR and the instruments.

The result of A3 will be a statement on requirements. Other aspects, such as proposals for further actions, descriptions of the client situation or specifications of certain processes, are also part of the report assuming they were among the objectives specified in the study plan.

It is essential that the analyst assesses these findings especially the elaborated requirements in the light of the two previous activities, i.e. whether the answers he gives are really the ones to solve the problems identified in activity A1, and whether they observe the constraints and restrictions relevant to the problem situation. In particular, the answers have to be checked to ensure that they fulfil the demands of the study brief, (e.g. the objectives of the study) and of the study plan because these documents have been discussed with the client and were the basis of agreements and contracts.

Activity A4: Evaluation

In activity A4, the requirements identified in A3 are evaluated to *assess the benefits* to various stakeholders in the client situation. Depending on the individual conditions, a benefit analysis study may concern:

- the requirements for an office system in terms of an exploratory study without yet considering the fulfillment of these requirements by a specific office system or through specific changes;
- a planned office system by comparing the projected situation with the current situation or by comparing different planned alternatives;
- an implemented system aiming at the identification of deviations from the plan and the reasons for this deviation.

The primary objective of the evaluation is to obtain information about the requirements from a different view by assuming that changes have already been made. It also tries to see the effects generated by these changes as well as their effects on both the achievement of objectives and the office workers using a system.

7. Conclusions

The FAOR Approach has shown itself to be a comprehensive - if perhaps complex - vehicle for eliciting office system requirements. Given the substantial number of office systems methodologies available (with FAOR being only one), potential adopters are faced with the difficult decision of which one to choose. An evaluation of the alternative approaches shows that the FAOR Approach compares favorably to other methodologies on most criteria. It is strong on dealing with the social domain. In particular, it has a strong 'unstructured task capability' and allows 'multiple perspectives' to be included in requirements analysis. Its value appears also high on the economic side, where 'cost-benefit analysis' support is called for. In terms of usage criteria, however, some FAOR's weaknesses begin to become manifest. In particular, it is weak with respect to criteria, such as 'understandability' and 'transferability'. Moreover, because of its newness, there is currently little in the way of 'computer support'. On the other hand, it is designed to be able to deal with a wide 'range of projects' with 'multi-criteria objectives', and its 'controllability', 'completeness' and 'flexibility', make it desirable for use in office systems analysis.

There are a number of further developments or enhancements to FAOR which would be desirable to undertake in the future. These basically fall into two categories: automated tools and additional instruments. The desirability of computer support for FAOR is clear. As there are a number of tasks associated with requirements analysis which lend themselves to computer assisting, the development of automated tools for FAOR is a key future enhancement. Although FAOR includes five instruments within its arsenal, there is no reason to assume these instruments are sufficient for requirements analysis nor complete. Whilst the instruments have so far proved to be acceptable in the cases undertaken, it seems highly likely that additional instruments would prove beneficial - particularly in complex situations.

Beside any extensions, however, FAOR needs to develop in terms of application experience which would allow one to better judge the usefulness of the various concepts as well as the means to further develop them.

References

- Bjerknes, G., Bratteteig, T. (1985)
Florence in Wonderland - Systems Development with Nurses.
 Paper presented at the Conference on Development and Use of Computer-based Systems and Tools, Aarhus August 1985.
- Checkland, P. (1972)
 Towards a Systems-Based Methodology for Real-World Problem Solving.
 In: *Journal of Systems Engineering*, Vol. 3, No. 2.
- Hirschheim, R. (1985)
Office Automation: A Social and Organizational Perspective.
 John Wiley & Sons, Chichester.
- Hirschheim, R.; Feeny, D. (1986)
 User Experiences with Office Automation: Some Lessons and Recommendations.
 In: *Journal of General Management*, Vol. 12, No. 2, Winter 1986.
- Hirschheim, R.A. and Smithson, E. (1987)
 A Critical Analysis of Information Systems Evaluation.
 to appear in: *Challenges in Information Systems Assessment*, edited by N. Bjørn-Andersen and G.B. Davis, Amsterdam 1987.
- Krückeberg, F. (1983)
 Bürokommunikation und ihr Umfeld.
 In: *Informationstechnik und Bürosysteme*, edited by P. Wisskirchen *et al.*, Stuttgart.
- Lanzara, G., Mathiassen, L. (1985)
 Mapping Situations within a Systems Development Project.
Information and Management, Vol 8, No. 1.
- Legge, K. (1984)
Evaluating Planned Organizational Change.
 London.
- Mumford, E. (1983)
Designing Human Systems.
 Manchester.
- Oppelland, H.J., Kolf, F., Claus, J. (1977)
Dokumentation der Ergebnisse einer Expertenbefragung zur Entwicklung und Einführung rechnergestützter Informationssysteme.
 PORGI-Projektbericht No. 5, Cologne (BIFOA).
- Pava, C. (1983)
Managing New Office Technology: An Organizational Strategy.
 Free Press, New York.
- Petri, C.A. (1980)
 Introduction to General Net Theory.
 In: *Net Theory and Applications* edited by Brauer, W. (Lecture Notes in Computer Science 84), Springer, Berlin *et al.*
- Schäfer, G. (1987a)
 Benefit Analysis of Office Systems: Concepts for a Method.
 to appear in *Challenges in Information Systems Assessment* edited by Bjørn-Andersen, N. and Davis, G.B. North Holland, Amsterdam.
- Schäfer, G. (1987b)
Functional Analysis of Office Requirements: A Multiperspective Approach.
 Publication in preparation.
- Welke, R. (1983)
 IS/DSS: DBMS Support for Information Systems Development.
 In: *Data Base Management Systems*, ed. by C. Holsapple and A. Whinston, D. Reidel, Amsterdam.
- Wood-Harper, A.T., Antill, L., Avison, D.E. (1985)
Information Systems Definition: The Multiview Approach.
 Blackwell, Oxford *et al.*

Project No. 1030

HUMAN, ORGANISATIONAL AND ECONOMIC (HOE) FACTORS IN IT UPTAKE PROCESSES
IN COMPLEX WORK ENVIRONMENTS (IT-UPTAKE)

Ryan, G., Cullen, K., Wynne, R., Ronayne, T., Cullen, J., Dolphin, C.

*Work Research Centre, Psychosomatic Unit, Garden Hill, 1 James's St.,
Dublin 8, Ireland*

Korte, W.B., Robinson, S., (*Empirica, FRG*)

Ennis, B., Hopkins, M. (*Memory Computer, Ireland*)

IT-UPTAKE focuses upon the role of human, organisational and economic (HOE) factors in the effective and productive uptake of IT application systems in complex work environments (CWEs). The project explicitly addresses IT uptake both in the more traditionally organised work environments and in the new forms of work organisation made possible by teleworking. While office systems are the major focus of concern, IT-UPTAKE also addresses developments in Computer Integrated Manufacturing. The major outputs from IT-UPTAKE are a generic model of human, organisational and economic (HOE) factors in IT uptake processes, an instrument for the collection of information concerning these HOE factors in real working environments, field trials of the generic model and instrument in real working environments, and guidelines concerning the management of HOE factors in the uptake of IT in real working environments. The paper outlines project results and achievements concerning the development, field trialling and refinement of the generic model and its associated instrument, and the development of guidelines. Packaged guidelines and a packaged final report on the project are expected in 1988. The project is complementary to ESPRIT Project 56: (FAOR) Functional Analysis of Office Requirements.

1. INTRODUCTION

ESPRIT Project 1030 (IT-UPTAKE) is concerned with the investigation of those human, organisational and economic (HOE) factors which facilitate or constrain the uptake of IT in complex working environments (CWEs), in which teleworking is playing an increasingly important role. These factors have major implications for the rate, extent and consequences of IT utilisation and exploitation within the European Community.

The core concern of Project 1030 involves the application of *human, organisational and economic perspectives* to the uptake of IT within complex work environments. Each of these three core facets is discussed briefly below. The explicit coverage of both the more traditionally organised work environments and the new forms of organisation made possible by teleworking is also discussed.

First, the focus of Project 1030 concerns the uptake of IT within particular workplaces. In this sense it is viewed as adding a perspective complementary to that of the many studies of the diffusion of IT across organisations [e.g. 1,2, 3,4,5]. While some of these diffusion studies have examined factors associated with higher or lower diffusion rates (e.g. industrial sector, size of enterprise, geographical location and type of ownership) little attention has been focused on those human, organisational and economic factors - either internal or external to the organisation - which facilitate or constrain the process of IT uptake. As with parallel developments within the organisational innovation research tradition in the U.S. [e.g. 6,7], the project emphasises the importance of the innovative technology, the context into which it is being introduced and, especially, the process of embedding the technology in its context. The IT uptake process refers to this embedding process and the project looks at the experiences of organisations involved in this process.

Another limitation of diffusion studies is the lack of coverage of such issues as level of usage of IT applications within the workplace, whether such applications are used optimally or efficiently, or the human, organisational, and economic consequences of the uptake of IT. The project specifically addresses these aspects of the IT uptake process.

Second, the project focuses on the uptake of IT within the workplace rather than other locations. The workplace is becoming an increasingly important arena for the uptake of IT applications, as instanced, for example, in the best selling applications and utility software in the U.S. [8]. Also, the development of cheaper, smaller and more powerful IT products, coupled with the development of sophisticated communications and networking facilities is shifting the locus of IT uptake within organisations away from technically specialised departments (traditionally DP departments) to many other contexts within organisations [9, 10]. Arising from this there emerges an increasing organisational complexity associated with IT uptake in the workplace [11]. The increasing uptake of both standalone IT applications and applications allowing decentralised access to centralised DP department facilities are both important in this regard. The concept of *complex work environment* refers to these increasingly complex situations within which IT uptake occurs in the workplace.

Third, *human, organisational and economic factors* arising in the IT uptake process in CWEs need to be addressed for a number of important reasons. For purposes of Project 1030 *human factors* refer to "... all aspects of the relation between the human, the machine and the environment which directly influence the safe, efficient, acceptable and satisfying usage of the IT system" [16]. *Organisational factors* expand the relatively narrow focus of traditional ergonomics to cover those aspects of complex organisations (e.g. structure, process, climate/culture) which may influence and, in turn, be influenced by the IT uptake process. *Economic factors* refer to those economic considerations which may influence the IT uptake process and those economic criteria which may be brought to bear in assessing the outcome of IT uptake processes in CWEs.

At least five factors suggest the importance of these perspectives:

. The slower than anticipated uptake of IT in the workplace [12]

- . The increasing competitive importance of the incorporation of human factors considerations in IT products and systems [13,14,15,16,17]
- . Evidence of failure of many IT systems in CWEs [18]
- . Evidence of much less than optimal use/exploitation of IT systems installed in CWEs [18]
- . Concern about the social effects and quality of working life implications of IT systems in CWEs [19,20,21,22,23].

The first four aspects refer primarily to economic concerns and the fifth involves social concerns. To the extent that human and organisational factors influence the first four, and the economic imperative influences the fifth, it will be clear that human, organisational and economic factors in the IT uptake process are interlinked in a complex manner. Consequently, the simultaneous application of these complementary perspectives can make an important contribution to the attainment of both economic and social goals.

Finally, the project includes *specific coverage of teleworking* as a new dimension of CWEs involved in IT uptake. The advancing development and application of IT will lead to major changes in the European and world labour markets in the coming decades. This technology has already made the dispersion of certain areas of office work over time and space possible; and, in the future, the more these and other office functions become supported by computer systems and the more computers become linked by communication networks, the less it will be necessary for office workers to be located in the same office or even the same building [24].

Teleworking must be seen as an integral option in an IT uptake process; it represents a major driving force behind the increasing pace of IT uptake processes extending into the future [25]. IT uptake may be seen as both a prerequisite for, and a consequence of, the adoption of the teleworking option. The increasing importance of teleworking as an IT uptake arena is addressed within the project. Consequently, IT uptake processes are addressed within both the more traditional forms of work organisation and the newer forms of work organisation made possible by teleworking.

2. OBJECTIVES OF IT-UPTAKE

The objectives of this project are to provide conceptual frameworks and operational procedures for the characterisation of IT uptake processes in terms of HOE perspectives. These profiles of IT uptake processes, in turn, enable the formulation of guidelines concerning the uptake of IT in potential and actual user environments. These guidelines are based on the development of models of uptake processes, which are formulated and tested in real work settings during the project. The guidelines which are emerging from the work of the partners will be in a format that is relevant to a variety of office and other sectors where IT systems are potentially or actually applicable on a wide scale.

IT-UPTAKE examines the role of HOE factors in the uptake of IT application systems in three working sectors: office, health care and manufacturing. Teleworking situations may arise within all of these sectors. The specific objectives of the project are as follows:

- . Identifying and investigating the role of HOE factors in the various working contexts;

- . Development of a conceptual framework for characterising IT uptake processes within complex working environments;
- . Development of methodologies for investigating the role of HOE factors in the uptake of IT in the various working contexts;
- . Highlighting the importance of ensuring that human, organisational and economic factors are given appropriate consideration at all stages in the development and implementation of IT in complex working environments; and
- . Formulating guidelines concerning strategies for the management of HOE considerations arising throughout different stages in the uptake of IT in CWEs.

3. THE PARTNERS AND THEIR DOMAINS OF EXPERTISE

Four organisations are working together to achieve the objectives of IT-UPTAKE:

- . The Work Research Centre (Dublin, Ireland) contributes expertise regarding human and organisational factors in Information Technology;
- . Empirica (Bonn, FRG) contributes expertise concerning HOE factors and the technological infrastructure affecting the diffusion potential of telework;
- . Memory Computer/Irish Medical Systems Ltd. contributes expertise concerning traditional business systems analysis techniques; and
- . STL (UK) provides links with the complementary ESPRIT Project 56 concerning functional analysis of office requirements (FAOR).

The Work Research Centre acts as Technical Manager for the project. Memory Computer acts as Prime Contractor for the project. The four partners are actively collaborating in the realisation of the overall objectives of IT-UPTAKE.

3.1 Objectives of the Human and Organisational Factors Dimension of IT-UPTAKE

Human and organisational factors issues arising in the IT uptake process in CWEs are the major concern of the Work Research Centre within Project 1030. The objectives of the human and organisational factors dimension of IT-UPTAKE are:

- . To identify and characterise the range of human and organisational factors that may facilitate or constrain the effective and productive uptake of IT application systems in end-user organisations;
- . To develop a framework for the systematic conceptualisation and analysis of these human and organisational factors;
- . To specify instruments that may be used for the generation of quantitative and/or qualitative information concerning the role of these human and organisational factors in the IT-uptake process;
- . To generate information concerning the effects of those human and organisational factors that actually arise in the context of IT-uptake processes occurring in the working environments currently being investigated within the project; and
- . To generate guidelines that may be used by a wide range of key actors for the management of human and organisational factors in IT uptake within CWEs.

The importance of understanding the nature and contribution of human and organisational factors in the development, implementation and operation of IT products and services is now clearly recognised. However, alongside the increasing recognition of these sets of factors there is a growing awareness of the need to organise existing knowledge in a systematic way and to develop strategies

for the effective transfer of this knowledge to key groupings such as the European IT industry, vendors of IT products and services, end-user organisations and individuals. IT-UPTAKE is actively addressing these needs and has already generated technical products that are of considerable relevance for the understanding and management of HOE factors in the development, implementation, and operation of IT application systems within end-user organisations particularly.

3.2 Objectives of the Teleworking Dimension of IT-UPTAKE

The objectives of the teleworking dimension are to investigate the supplies of and demand for teleworking arrangements, as well as the characteristics of the technology infrastructure which are a precondition for take-up in many applications. This investigation results in a profile of potential for telework inception involving IT uptake in CWEs in Europe.

Telework uptake takes place 'naturally' in CWEs because of the advantages perceived by potential participant teleworkers and teleworking organisations [26]:

- . telework offers a wide range of options with regard to flexibility in the locational as well as the temporal performance of office work to employers, employees and other individuals [27,28,29];
- . uptake is also driven by certain economic imperatives (e.g. increase in productivity, reductions in overhead expenses) which are of major interest to employers as well as individuals intending to start their own business [30, 31,32]; and,
- . telework uptake is furthered by specific trends currently taking place in society and also by changes in working patterns (e.g. increases in subcontracting and self-employment, externalization of labour) [25,33].

Given the still modest but growing and potentially ubiquitous implementations of telework and its importance for IT uptake, a sound empirical basis for the judgement of its potential and for the identification of facilitating/constraining factors is required. Available information about the state of the art and future development of teleworking is largely speculative; the little empirical work available is to a large extent confined to the geographical and social conditions found in the U.S.A. Within IT-UPTAKE a substantial body of systematic information concerning telework uptake within the member states is being generated and makes an important contribution to the development of the model, instrument and guidelines.

3.3 Objectives of the Vendor/Traditional Business Systems Analysis Dimension of IT-UPTAKE

Among important factors influencing the uptake process are those related to the vending of IT systems and to the systems analysis, design, and implementation of IT systems. These are therefore addressed by a commercial partner with expertise in these areas.

The aims of this section of the project are to:

- . Provide an economic/commercial perspective on the factors involved in the uptake process;

- . Incorporate factors associated with the vending process into the model of the uptake process;
- . Incorporate factors associated with the systems analysis and design process into the model of the uptake process;
- . Provide a human and organisational factors perspective on the traditional systems analysis and design process;
- . Provide a commercial perspective on the possibilities of using telework in test bed site(s); and
- . Apply a model of the uptake dynamics of telework to test bed site(s).

The project views the integration of behavioural and social science expertise of the research partners with the practical systems analysis and commercial expertise of the commercial partners as especially important. This component of the project makes a major contribution in this regard.

3.4 Objectives of the FAOR Dimension of IT-UPTAKE

FAOR offers to systems analysts an innovative approach aimed at providing an analysis method for the office domain. It focuses on the early stages of the life cycle of an office system, where the determinants of a successful application of office technology have to be identified and translated into requirements for the application. [34,35]. However, introducing an office system is not purely, nor even mainly a technical issue. Its widespread implications for the way office work is carried out, make it to the same extent an organisational and social issue as well. An approach for analysing offices has to perceive these factors in their inter-relationships and to identify requirements suitable for the organisation as a whole and not merely its technical aspects.

The approach offered by FAOR for the analysis of offices has two foci for understanding: analysis - how analysis should be carried out, and understanding the office - how is office work properly conceptualised. The FAOR approach develops a multi-perspective approach to analysis and a functional approach to office work. The multi-perspective approach means that the systems analyst explicitly acknowledges his or her hermeneutic stance as an observer and interpreter, and methodically recruits and integrates a range of perspectives - complementary and conflicting - into the analysis. When it comes to specifying requirements, the ambiguity of coexisting interpretations due to different perspectives and the various facets of the problem situation has to be resolved. The vehicle for dealing with this in FAOR is through focusing on the functionality of requirements. The functional approach distinguishes between the basic functions of office work for the organisation, and any particular implementation of those functions in terms of current office activities. A functional analysis seeks to abstract the basic functions of office work, and not to extract a contextually specific set of office activities. A functional analysis specifies requirements in terms of the organisational functions of office work, and not in terms of automating a current set of office activities [36].

FAOR has developed tailorable instruments to provide the practical means for applying perspectives. These include the Function analysis instrument, the Communication analysis instrument, the Information analysis instrument and the Benefit analysis instrument, some of which are currently being considered for tailoring and use in the field trialling activities of IT-UPTAKE.

3.5 Integration

Together, the four components of the project outlined above provide the necessary framework for the integration of the domains of expertise of the four partners. This resource framework is currently being integrated to produce an extended version of the model and instrument for application in collaborative field trials involving all the partners. In this way the four domains of expertise (human and organisational factors in IT uptake processes; traditional business systems analysis/vendor perspectives; the diffusion potential of telework; and relevant aspects of the functional analysis of office requirements) are being integrated in model/instrument development and field trialling, and will provide considerable breadth and richness in the ensuing guidelines.

4. MAJOR PRODUCTS BEING GENERATED WITHIN IT-UPTAKE

The partners within IT-UPTAKE are collaborating in the generation of a number of key technical products from the project. These technical products comprise:

- . A model or framework for developing an understanding of the range of HOE factors of relevance to the uptake and implementation of IT in CWEs;
- . An instrument or set of operational procedures to allow the systematic collection of information concerning HOE factors relevant to IT uptake in CWEs;
- . Field trial reports concerning the development and implementation of IT systems in CWEs, using the model and instrument within real working environments;
- . Guidelines concerning the management of those HOE factors that arise throughout the various stages in the uptake of IT in work environments. These guidelines are to be targeted towards key actors involved with the uptake of IT in CWEs.

The model is primarily intended to serve as a vehicle for communicating a conceptualisation of the IT uptake process and of the role of HOE factors in this process within CWEs. The instrument comprises an organised set of qualitative and quantitative procedures for gathering information within CWEs. The various sub-components of the instrument allow the systematic generation of information concerning the interaction of HOE factors with the IT process within CWEs. The instrument allows the operational application of the model in *field trials* in actual CWEs. These field trials provide the opportunity to evaluate and refine the model and instrument. They also provide important input to the development of the guidelines. The *guidelines* are intended for use by a number of actor groupings involved in the IT uptake process. In particular, attention will be paid to the needs of key actors in end-user organisations (e.g. management, project teams, DP professionals, end-users and their representatives), vendors and suppliers of IT products and services, and IT manufacturers.

The Model, Instrument, Field trials and Guidelines constitute a substantial body of systematic information concerning HOE factors in IT uptake processes located within work environments. The preparation and packaging of guidelines for the promotion of the optimal uptake of IT applications in these CWEs represents a key technical output from IT-UPTAKE. In addition to packaged guidelines, IT-UPTAKE is generating a comprehensive model for understanding human, organisational and economic factors in the uptake of IT application systems in CWEs. The instrument comprises a set of procedures enabling the operational-

isation of the model within specific contexts of IT uptake. Currently, the partners are considering possibilities for generating packaged versions of both the model and instrument, in addition to the packaged guidelines being developed within IT-UPTAKE. Finally, the field trials and test bed site investigations being undertaken by the partners within IT-UPTAKE are generating a significant body of information concerning HOE factors in IT uptake processes occurring within CWEs. These field trials enable the evaluation and refinement of the model and instrument and also facilitate the formulation of guidelines targeted towards IT uptake activities and processes occurring within CWEs.

5. CURRENT STATUS AND ACHIEVEMENTS

First versions of the model and instrument have been developed and subsequently refined through field trialling in actual CWEs. The refined version of the model provides a framework for conceptualising, understanding and, ultimately, investigating or intervening in the IT uptake process. This model presents the IT uptake process as a particular form of organisational change which may or may not be initiated depending on the configuration of a variety of forces internal and external to the organisation. The change processes arising through IT uptake are viewed from three perspectives : human, organisational and economic. The influence of factors along these three dimensions on the uptake process and, conversely, the implications of the uptake process for organisational performance along the three dimensions are addressed by the model. The refined version of the instrument provides an organised pool of instrumentation for information collection in actual complex work environments. The instrument is structured according to the refined version of the model and provides procedures for tailoring instrumentation for use in actual CWEs.

An analysis of the potential and the uptake dynamics of telework has been conducted through large scale empirical surveys and refinement of their indications through detailed field trials. The widely based representative surveys cover the appropriate technical, supply and demand aspects allowing the description of the resource framework for telework as a new form of work organisation. The field trials in existing teleworking schemes provide a more detailed overview of the characteristics of current schemes from which the findings from the representative surveys may be evaluated, refined and expanded. A particularly important output has been the development of a first version of a model of the uptake dynamics of telework. This provides an important basis for the development of the extended model of HOE factors in IT uptake processes, to include both the more traditional forms of work organisation and the newer forms facilitated by telework.

A test bed investigation has been undertaken using traditional business systems analysis techniques and an IT system has been proposed which would support and improve the functioning of the test bed site. The significant features of this proposed systems design concern the systematic incorporation of human, organisational and teleworking perspectives into the final design. This was achieved through the application of the model of HOE factors in IT uptake through the usage of its associated instrument in the test bed site. Consequently, the analysis and design process was conducted in a manner to take systematic account of human and organisational factors and the final design reflects human and organisational factors of relevance in the test bed site. Also, the developing model of the uptake dynamics of telework has been applied in the test bed site and potential telework applications identified.

6. SIGNIFICANCE OF ACHIEVEMENTS/ACTUAL OR POTENTIAL INDUSTRIAL IMPACT

A key objective of ESPRIT is to provide European IT industry with the basic technologies it needs to be competitive on the world market in the medium to long range, in an enduring manner. In this context, the objectives of IT-UPTAKE are of relevance to ESPRIT Projects which are more directly concerned with the provision of basic technologies for the European IT industry. IT-UPTAKE focuses upon the investigation of those HOE factors which facilitate or constrain the uptake of IT in complex work environments. These 'non-technological' factors have major implications for the rate, extent and consequences of IT uptake within end-user organisations throughout the European Community. [37,38]

Several recent investigations have suggested that the quality of the management of the introduction of the new IT application systems into CWEs requires considerable improvement [39,40,41]. For instance, in a recent study of current practices in the evaluation of those new technologies being introduced into British organisations [39], it is concluded that the present standard of management of the uptake and implementation of the new technologies is not high and the need for a wider appreciation of approaches from the behavioural sciences to the management of technological innovation within real working environments is emphasised.

The model, instrument, and guidelines currently being developed within IT-UPTAKE are of considerable relevance in this context in so far as these activities involve collaboration with the variety of professional groups and other key actors involved with the development and implementation of IT application systems within CWEs. Already, within IT-UPTAKE collaborative working relationships have been established between specialist professional groups within the behavioural sciences and researchers/practitioners concerned with the design, development and implementation of information technology application systems in real working environments. These collaborative relationships have been established in two inter-related contexts: (i) between the industrial and research partners within IT-UPTAKE (the "Collaboration Strategy") and (ii) between the partners within IT-UPTAKE and representatives of the wide range of CWEs accessed for purposes of the project (the "Complex Work Environment Strategy"). The "Complex Work Environment Strategy" provides access to an important variety of test-bed sites for the design, development, implementation, evaluation and refinement of the major outputs currently being generated within IT-UPTAKE. This "Complex Work Environment Strategy" has involved the development of dialogue with a wide range of relevant actors in actual complex working environments where IT uptake processes are actually occurring or are projected to occur.

In terms of industrial applications of the products being generated within IT-UPTAKE, these collaborative relationships provide a means for ensuring that considerations of demonstrated practical significance in the uptake of IT application systems are incorporated into the developing Model, Instrument, Field Trials and Guidelines. Arising from these collaborative relationships, a forum has been generated whereby relevant actors within end-user organisations may begin to conceptualise and examine in an integrated fashion HOE factors as they arise in IT uptake processes occurring within their own organisations. In this sense, IT-UPTAKE is already exerting some influence on the uptake of IT application systems within those working environments accessed for purposes of this project.

IT-UPTAKE will provide the European IT industry, vendors/suppliers of IT products and services and end-user organisations with a framework for characterising the range of activities and processes associated with the development and implementation of IT application systems within complex working environments. This framework will provide the European IT industry with conceptual and operational procedures for the characterisation of IT uptake processes. The model (s) of the processes associated with the uptake of IT application systems in real working environments and the associated instrument and guidelines currently being developed, evaluated and refined within IT-UPTAKE will provide the European IT industry with a set of strategies for the effective and productive management of technological change programmes in a variety of industrial and service sectors.

7. EXISTING/POTENTIAL APPLICATIONS

The Model, Instrument, Field Trials and Guidelines issuing from IT-UPTAKE are already generating a substantial body of systematic information concerning HOE factors in IT uptake processes located within two sectoral environments, specifically: office and health care. In addition, it is proposed to extend these analyses to IT uptake processes located within manufacturing environments. Teleworking applications may arise or be appropriate in all of these sectoral contexts. A key technical output from IT-UPTAKE concerns the preparation and packaging of guidelines for the promotion of the optimal uptake of IT applications in these CWEs. In addition to economic considerations, these guidelines will focus specifically upon strategies for the effective and productive management of the range of human and organisational considerations that arise throughout different stages in the uptake of IT in complex work environments.

There is now increasing research and commercial interest in developing an understanding of the IT uptake process in CWEs. The guidelines currently being developed within ESPRIT Project 1030 will focus upon the elaboration of strategies for the management of these human and organisational factors in IT uptake processes, in addition to economic considerations. In terms of the objectives of ESPRIT, the formulation of these guidelines will facilitate the transfer of knowledge concerning human factors and organisational processes in IT uptake to the European IT industry and European organisations engaged in IT systems development and implementation activities. The formulation of such guidelines will also highlight priority areas for the development of approaches from the behavioural and management sciences to the uptake and implementation of IT in complex working environments. Several commentators have recently emphasised the importance of incorporating approaches from these disciplines for ensuring that the basic technologies being developed within programmes such as ESPRIT are optimally exploited throughout both the European IT industry and end-user European organisations in particular.

The Model and Instrument being developed within IT-UPTAKE also provide methodologies and tools for monitoring and evaluating the contribution of HOE factors to IT uptake processes in end-user organisations. In this context, IT-UPTAKE addresses in a systematic manner the HOE factors that may facilitate or constrain the optimal uptake and implementation of IT application systems within European end-user organisations. Several commentators have emphasised the need for evaluating the HOE implications of the new IT applications for working life within the member states [19,20,21,37,38,39,40,42]. Engagement by end-user organisations in systematic monitoring and evaluation of HOE factors in IT uptake processes has been significantly hampered by a perceived lack of rel-

evant and accessible conceptual frameworks and their associated methodologies for the evaluation of these aspects of IT uptake processes [39]. The model and instrument currently being developed by the consortium of partners within IT-UPTAKE, in combination with the products already generated within the complementary ESPRIT Project 56, represent an important contribution to the European IT industry and European end-user organisations in this context.

8. TIME HORIZON FOR INDUSTRIAL APPLICATIONS

The four major products being generated by the partners within IT-UPTAKE will be available in their final formats in April, 1988. These products comprise packaged guidelines, a model of HOE factors in the IT uptake process, an instrument (or set of instruments) and reports concerning the range of field trials and test bed site investigations.

As ESPRIT Project 1030 progresses, procedures will be developed to encourage the European IT industry, together with organisations that are vendors and end-users of IT application systems, to obtain maximum benefit from the products being generated within IT-UPTAKE. The partners within IT-UPTAKE are exploring possible programs of activity that may be used for this purpose. These include the organisation of conferences, seminars and workshops; and the provision of consultancy, research and training services targeted towards the transfer of knowledge concerning HOE factors in IT uptake processes to the European IT industry and organisations that are end-users of European IT products. In this context, discussions with other related ESPRIT projects have already been initiated. In particular, ESPRIT Projects such as Project 385 (HUFIT) address issues of considerable relevance for the dissemination of the products being developed within Project 1030.

ACKNOWLEDGEMENTS

The partners in Project 1030 would like to acknowledge the support of the Commission of the European Communities as represented by J. Roukens and J. Machnik of the Office Systems Group.

REFERENCES

- [1] Gillespie, A., Godard, J., Robinson, F., Smith, I. and Thwaites, A., The effects of new information technology on the less-favoured regions of the community. Studies Collection: Regional Policy Series No. 23. (Office for Official Publications of the European Communities, Luxembourg 1985).
- [2] McAndrew, M. and Mulhall, J., Office automation in Irish business. A report by the Irish Management Institute. (Irish Management Institute, Dublin, 1986).
- [3] Northcott, J. and Rogers, P., Microelectronics in Industry: what's happening in Britain. (Policy Studies Institute, London, 1982)
- [4] Northcott, J., Rogers, P., Knetsch, W. and De Lestapis, B., Microelectronics in Industry. An International Comparison: Britain, Germany, France. (Policy Studies Institute, London, 1985).

- [5] NSC, First annual survey of computer usage in Ireland. (National Software Centre, Ltd., (NSC), Dublin, 1985)
- [6] Bikson, T.K. and Eveland, J.D., *New Office Technology: Planning for People. Work in America Institute; Studies in Productivity: No. 40.* (Pergamon Press, New York, 1986)
- [7] Tornatzky, L.G., Eveland, J.D., Boylan, M.G., Hetzner, E.C., Roitman, D., and Schneider, J., *The process of technological innovation: reviewing the literature.* (National Science Foundation, Washington, D.C., 1983)
- [8] OECD, *Information Computer Communications Policy No. 9: Software An Emerging Industry.* (OECD, Paris, 1985)
- [9] Jarke, M. (ed.), *Managers, Micros and Mainframes: Integrating Systems for End-Users.* (Wiley, Chichester, 1986)
- [10] Keen, P., *The VDT as an agent of change*, in: Bennet, J., Case, D., Sandelin, J. and Smith, M. (eds.), *Visual display terminals: usability issues and health concerns.* (Prentice-Hall, Englewood Cliffs, 1984).
- [11] Ballou, D. and Kim, S., *A systems life cycle for office automation projects.* *Information and Management*, 1984, 7, 111-119
- [12] Faehnrich, K., Fauser, A. and Heller, N., *The extent of introduction of electronic machinery in the office.* Report for the European Foundation for the Improvement of Living and Working Conditions (Dublin, European Foundation for the Improvement of Living and Working Conditions, 1984).
- [13] Christie, B., *Human factors of the user-system interface. A report of an ESPRIT preparatory study* (North-Holland, Amsterdam, 1985)
- [14] Davies, D., *HUFIT's role in office systems design*, in: Commission of the European Communities, Directorate General XIII, *Telecommunications, Information, Industries and Innovation* (eds.) *ESPRIT '86: Results and achievements* (Elsevier Science Publishers B.V. (North-Holland) for the Commission of the European Communities, Amsterdam, 1987).
- [15] Ericsson Information Systems, *Ergonomic principles in office automation: State of the art reports and guidelines on human factors in the office environment* (Ericsson Information Systems, Stockholm, 1983).
- [16] Shackel, B., *Ergonomics in information technology in Europe - a review.* *Behaviour and Information Technology*, (1985) 4 (4), 263-287.
- [17] Bjorn-Andersen, N., *User driven systems design.* *Work and People*, (1984), 10, 17 - 23.
- [18] Hirschheim, R.A., *Office automation: a social and organisational perspective* (Wiley, Chichester, 1985).
- [19] CEC, *Social Europe: Supplement on New Technologies and Social Change.* Office automation. CEC D-G for Employment, Social Affairs and Education (Office for Official Publications of the European Communities, Luxembourg, 1985).
- [20] CEC, *Information report of the Section for Social Questions on the New Technologies - Social Aspects* (CEC Social and Economic Committee. SOC/111 *New Technologies - Social Aspects.* CES 671/85 Brussels, 1986).
- [21] Bjorn-Andersen, N., Hedberg, B., Mercer, D. Mumford, E. and Sole, E. (eds.), *The impact of systems change in organisations: results and conclusions from a multinational study of information systems development in banks.* (Sijthoff and Noordhoff, The Netherlands, 1979)
- [22] FAST., *The FAST Programme. Volume 1. Results and recommendations.* (Commission of the European Communities, Brussels, 1982)
- [23] Kling, R., *Social analyses of computing: theoretical perspectives in recent empirical research.* *Computing Surveys*, (1980) 12 (1), 61-110.
- [24] Olson, M. and Tasley, R., *Telecommunications and the changing definition of the workplace.* Presentation to the Telecommunications Policy/Research Conference, Annapolis, Maryland, April 26, 1983.

- [25] Steinle, W. Opening remarks and introduction, in: Korte, W., Robinson, S., and Steinle, W., (eds.), *Telework - present situation and future development of a new form of work organisation* (North-Holland, Amsterdam), in print.
- [26] Kelly, M.M. and Gordon, G.E., *Telecommuting. How to make it work for you and your company* (Prentice-Hall, Englewood-Cliffs, New Jersey, 1986)
- [27] Brandt, S., *Aufgebendezentralisierung durch moderne kommunikations mittel* (CW-Publikationen, Munchen, 1984)
- [28] Diebold Group Inc. (eds.), *Office work in the home: scenarios and prospects for the 80's.* (New York, 1981)
- [29] Nilles, J.N., Carlson, S.I., Gray, B. and Hanneman, G.J., *The tele communications - transportation trade-off. Options for tomorrow.* (Wiley, New York, 1976)
- [30] Gordon, G., *The dilemma of telework - technology versus tradition*, in: Korte, W. Robinson, S. and Steinle, W. (eds.), *Telework - present situation and future development of a new form of work organisation* (North-Holland, Amsterdam) in print.
- [31] Judkins, P., *Networking - the Rank Xerox experience and its implications*, in: Korte, W. Robinson, S. and Steinle, W. (eds.), *Telework - present situation and future development of a new form of work organisation* (North-Holland, Amsterdam) in print.
- [32] Judkins, P., West, D. and Drew, J. (1985). *Networking in organisations* (Gower Press, Aldershot, 1985)
- [33] Huws, U., *Remote possibilities - some difficulties in the identification and quantification of telework in the U.K.*, in: Korte, W. Robinson, S. and Steinle, W. (eds.), *Telework - present situation and future development of a new form of work organisation* (North-Holland, Amsterdam) in print.
- [34] O'Donovan, P. (1986). *Functional analysis of office requirements (FAOR)*, in: *The Commission of the European Communities (eds.) ESPRIT '85: Status report of continuing work* (Elsevier Science Publishers B.V. (North-Holland) for the Commission of the European Communities, Amsterdam, 1986)
- [35] Schaefer, G. O'Donovan, P., Harper, M. Hansjee, R. and Domke, M., *FAOR: a multiperspective approach for analysing office system requirements*, in: *Commission of the European Communities, Directorate General XIII, Telecommunications, Information, Industries and Innovation (eds.) ESPRIT '86: Results and achievements* (Elsevier Science Publishers B.V. (North-Holland) for the Commission of the European Communities, Amsterdam, 1987)
- [36] *ESPRIT Project 56, Functional Analysis of Office Requirements (FAOR). Main Report (Draft), (1987).*
- [37] Otway, H.J. and Peltu, M. (eds.), *New office technology: human and organisational aspects. A publication from the INSIS programme of the Commission of the European Communities.* (Frances Pinter for the Commission of the European Communities, London, 1983)
- [38] Otway, H.J. and Peltu, M. (eds.), *The managerial challenge of new office technology. A publication from the INSIS programme of the Commission of the European Communities.* (Butterworths, London, 1983)
- [39] Blackler F. & Brown, C., *Current British practice in the evaluation of the new information technologies*, in: Debus, G. and Schroiff, H.-W. (eds.) *The psychology of work and organisation.* (Elsevier Science Publishers B.V. (North-Holland), Amsterdam, 1986).
- [40] Blackler F. & Brown, C. *Alternative models to guide the design and introduction of new information technologies into work organisations.* *Journal of Occupational Psychology*, (1986), 59, 287 - 313.
- [41] Vijlbrief, H.P.J., Algera, J.A. and Koopman, P.L., *Management of automation projects*, in: Debus, G. and Schroiff, H.-W. (eds.) *The psychology of work and organisation.* (Elsevier Science Publishers B.V. (North-Holland), Amsterdam, 1986).
- [42] Blackler F. & Brown, C., *Evaluation and the impact of information technologies on people in organisations.* *Human Relations*, (1985), 38 (3), 213-231.

Project No. 285

EVALUATION CRITERIA FOR THE ANALYSIS OF OFFICE DESCRIPTIONS
BASED ON PETRI NETS

V. De Antonellis and C. Simone
Dipartimento di Scienze dell'Informazione
Università di Milano
Via Moretto da Brescia 9 - 20133 MILANO (Italy)

In the ESPRIT Project - OSSAD #285, Petri nets are the basis for descriptions of how the office work is performed, both in the present office and in its possible alternatives. Given such descriptions, it is necessary to be able to evaluate them in order to compare alternatives among themselves and with the present state, and then choose the one to be implemented. In the paper we provide indications to perform qualitative and quantitative evaluations which can be derived from the analysis of the structural aspects of the nets.

1. DESCRIPTION OF OFFICE WORK: OPERATIONS SCHEMATA

In the OSSAD Project a model (called Descriptive Model) has been defined for the description of the accomplishment of office work, taking into account organizational structure and technological support [Con86]. The basic concepts of the model fall into two general classes: one concerns the organization of people and the other focuses on the work to be done. The fundamental concept concerning the organizational structure is the role, that is a specific qualification or charge characterizing actors in the office. Aggregations of roles, usually on the basis of shared responsibilities and/or the need for coordinated direction and control, are captured by the concept of (organizational) unit. As regards the description of the work to be done, the key concept is that of task, that is a portion of office work assigned to a role (or to a unit, if we consider a higher level of abstraction). A task can be seen as a set of coordinated operations, that is, of minor units of work to which resources (data/objects) and facilities (support technology) are properly associated.

Since the description of tasks focuses on the dynamics aspects of the office, it should represent the coordination of operations not only in terms of resource utilization, production and communication, but also in terms of the rules governing the behavior of roles in performing them (e.g. order between operations, synchronization with operations of other tasks, and so on). The formal description of a task which explicitly represents the dependency/independency relationships between the operations performed by a role in that task is called operations schema. Its graphical representation is based on Petri nets, presented in 1962 by C.A. Petri as a formalism to model processes.

Relevant features of Petri nets, that make them suitable to our purposes, are:

- to represent explicitly relationships of causal dependency/independency;
- to describe structure and behaviour of systems in which non-determinism and concurrency are essential characteristics (as it is the case of organizations and offices);

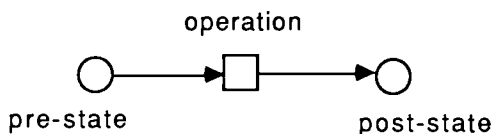
(as it is the case of organizations and offices);

- to combine the capability of explicitly representing operations (as several graphical/linear formal languages do) with the capability of explicitly representing states (as, for example, languages based on the notion of transition system do).
- to describe systems at different levels of abstraction;
- to verify properties of a system, like liveness, deadlock-freeness, etc., which contribute to prove its 'good behaviour'.

In our interpretation, a Petri net Π is defined as an ordered triple $\langle S, T; F \rangle$, where S is a set of states (of a role) and T is a set of operations (incorporated in a task of that role). F is a relation that relates: elements of S to elements of T (and, therefore, defines "pre-states" of operations) and elements of T to elements of S (and, therefore, defines "post-states" of operations).

The principal graphical representation is:

- states are represented by circles;
- operations are represented by boxes;
- their relationships are represented by arrows:

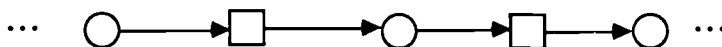


Associated to the above interpretation are rules for handling the dynamic behavior of the nets by means of the "markers game". The presence of a "marker" in a state, graphically a dot in a circle, indicates that the state holds, that is, the role is in that state and is allowed to execute one of the immediately following operations. In our interpretation, only one marker can be in a state. An operation is said "enabled" when all its pre-states hold, that is, it can occur. In this case, its pre-states are cleared out, while all its post-states become marked. It means that, after the execution of an operation, the role will be in the state immediately following that operation. In order to analyse the behavior, each net is given an initial marking (that is, a set of markers the presence of which is independent of any operation occurrence), then the markers game can start.

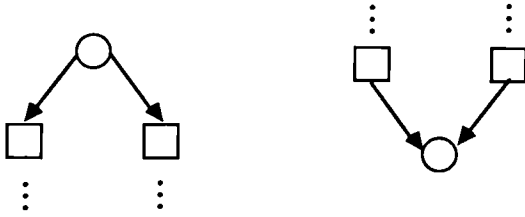
According to the previous definitions, an operations schema is represented by a graph composed of two types of nodes: states and operations connected by arrows in such a way that two nodes of the same type are not directly connected. Causal dependencies/independencies between operations are represented by:

- Sequence: operations executed in a predefined order:

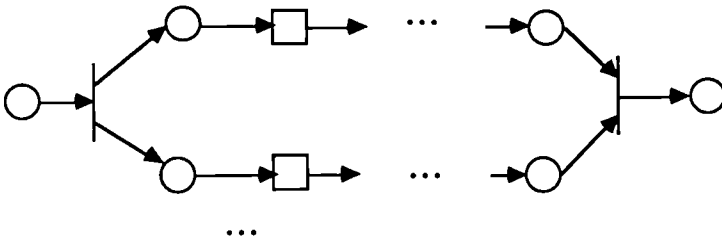
Ex.



- Conflict: mutual exclusion between operations which share the same pre(post)-states:
Ex.



- Concurrency: causal independency between operations, i.e. they can be executed in any order or simultaneously:
Ex.



(note: bars stand for merely structural operations, called empty operations, necessary to guarantee the correct dynamics of the net)

In the graphical representation, a net is labelled with the name assigned to the task it represents; each box is labelled with the name assigned to the corresponding operation, while bars remain unlabelled; circles are unlabelled. Note that morphological and syntactical rules have been defined to form names as expressions which are as complete and as unambiguous as possible, while preserving expressive power [Con86].

Loan Office

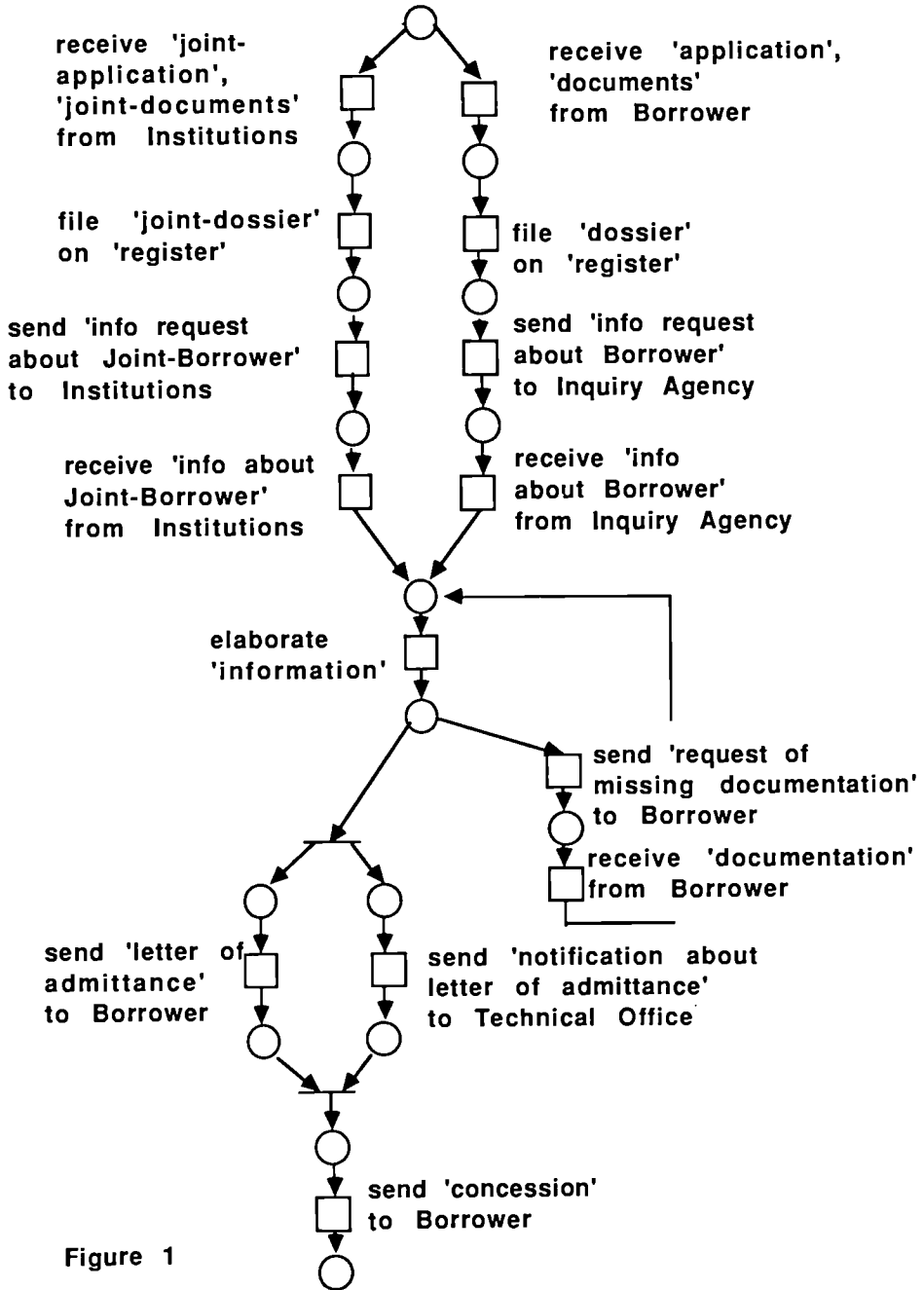


Figure 1

Since the operation names are expressions containing names of the involved resources, facilities and other roles or tasks, the net synthesizes all the considered aspects of the task: materials, products, instruments, communication and coordination.

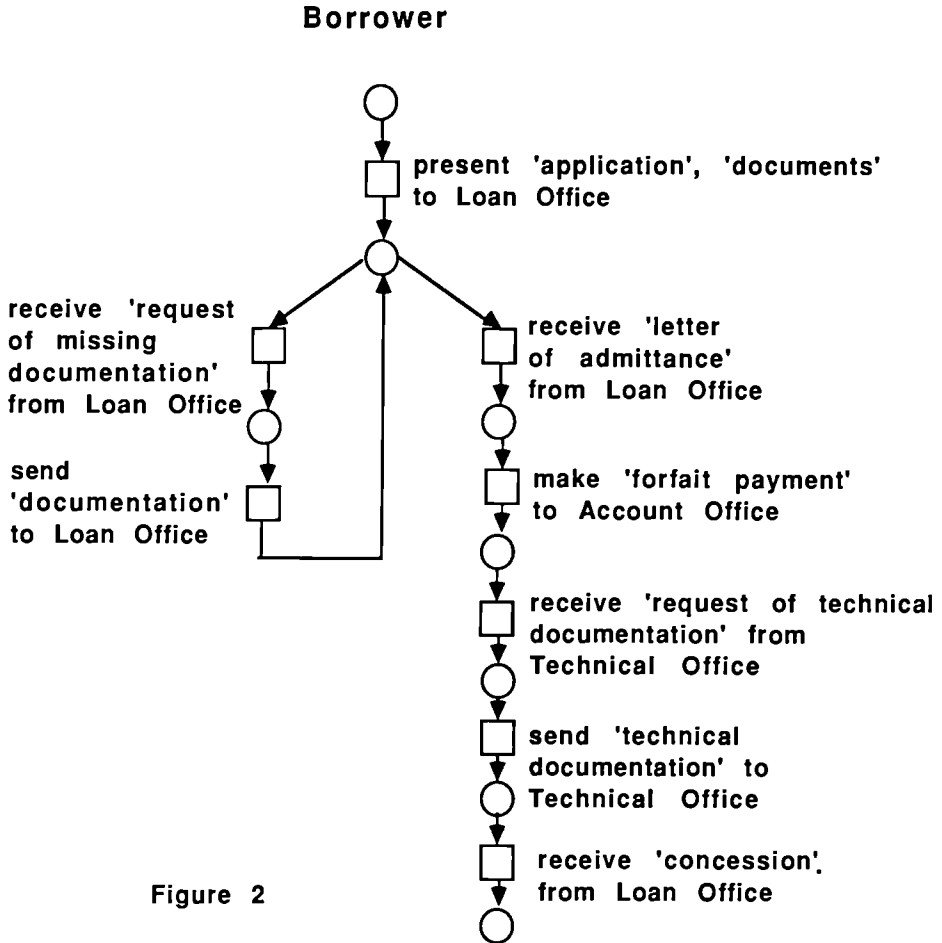


Figure 2

In figures 1 and 2 we give samples of operations schemata. Figure 1 represents the task performed by the unit Loan Office in an hypothetical bank for evaluating the opportunity of granting a loan requested either by a individual Borrower or by some Institutions, which are in charge of handling the loan for several individuals (Joint-Borrower). Figure 2 represents the task performed by an individual Borrower for obtaining a loan.

Loan Office

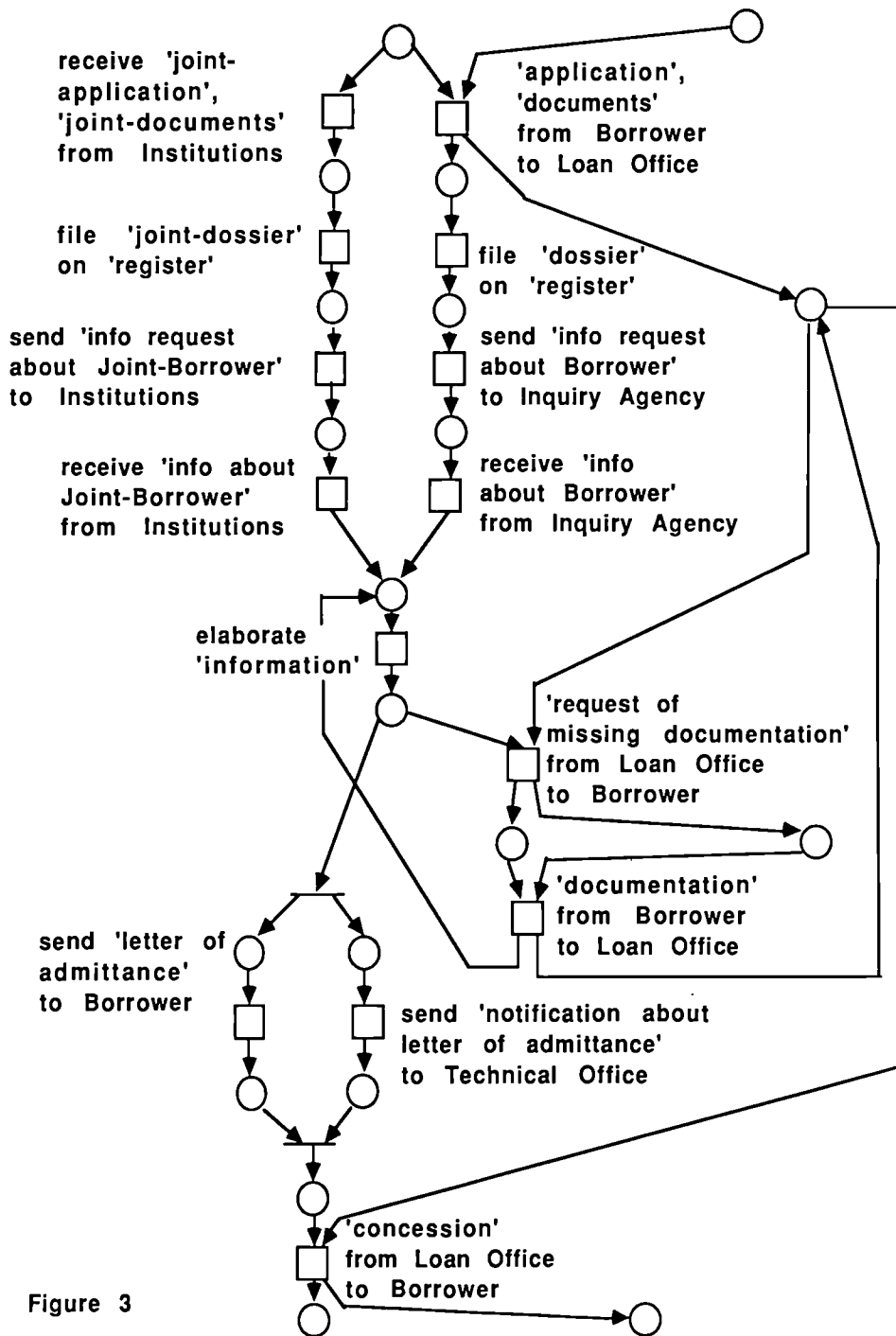


Figure 3

The operations schemata construction allows one to deal with small graphical descriptions without losing the relationships of the small portion with the remaining office representation. However, the nets corresponding to single tasks can be combined into major nets by merging some corresponding operations. The correspondence is defined at the interpretation level when operations indicating communication or cooperation between roles are recognized. The result is an operation which has as pre-states (post-states) the union of the pre-states (post-states) of the two sources operations with the related arrows. Figure 3 shows a combination of fragments of the nets in Figures 1 and 2, pointing out the interaction points which synchronize the two tasks.

The most general criteria of aggregation are:

- all the tasks pertaining to (part of) an activity: that is, a procedure;
- all the tasks pertaining to a role;
- all the tasks of tightly coupled roles.

Other aggregations of operations schemata can be conceived for specific purposes: for example, all the tasks using a facility, all the tasks accessing shared resources. They can be useful towards the specification of the technological support requirements.

In the following we provide some indications to perform qualitative and quantitative evaluations which can be derived from or suggested by the analysis of the operation schemata [Sim87]. Specifically, in this paper we will focus on how the type of relationship between operations, that can be concurrency, conflict or sequence, can be used as basis for evaluations. This is only one aspect of office evaluation, strictly related to efficiency and efficacy of operations and procedures, while in the whole OSSAD approach also parameters referring to the evaluation of the technological support (facilities), of the working environment and of the human factors are taken into account.

2. EVALUATION OF A SINGLE OPERATIONS SCHEMA

In the following, a systematic analysis of each type of relationship between operations is performed to point out relevant parameters for the evaluation of the execution of a task by the involved role.

Concurrency of operations

Inside a single operations schema this structure means that there is no predefined order as regards the execution of the operations by the involved role. This structure is specially useful to represent in a synthetic way the possible behaviour of several actors playing the same role. In fact, very often the order they give to the operations inside a task is inessential to understand the nature of the task, and can be taken into account at the level of human factors analysis.

The following parameters can be considered for evaluation:

- the amount and type of facilities/resources pertaining to the role and their availability. For example, one could recognize that an inadequate assignment or sharing constraints force an order in the actual execution. This case suggests to look for a possible

optimization (augmentation/reassignment) to improve flexibility in the execution.

- the amount and type of the concurrent operations. For example, in the case of many concurrent operations assigned to a role their execution results, actually, in an interleaving which can cause unpredictable delays, also w.r.t. the interactions with other roles. In this case, one could increase the efficiency and effectiveness of the execution by means of a reassignment of operations (on the basis of location of facilities, use of resources, and the like) to obtain a better specialization of the role.

Conflict of operations

This structure implies a choice among alternative operations. Possible considerations are related to the decisional process and the involved roles.

- Considerations on the decisional process derive from the joint analysis of the following parameters:

- modalities of conflict resolution, which can be:
 - . predefined by the organization (e.g., management policy) or by external regulations (e.g., laws, contracts, etc.);
 - . discretionary, e.g., according to the contingent situation;
- typology of the conflict, which can be:
 - . internal (local), if it is solved by the role under examination;
 - . external (global), if it is solved by other roles;
- constraints to the conflict resolution, which can be:
 - . resources availability;
 - . facilities availability;
 - . actors availability.

Evaluations can be made as regards:

- the adequacy of the criteria predefined by the management with respect to the actual working environment, by considering productivity aspects : effectiveness vs anarchy;

- the available support technology (e.g., information availability or communication facilities);

- the generation of waits, delays, other unproductive times, for example, in the interaction of the task under consideration with other tasks.

- Considerations on roles can derive from the analysis of the following parameters:

- degree of dependency of the role with respect to other roles on the basis of the ratio between local and global conflicts. This parameter can contribute to the evaluation of job satisfaction;

- possibility of refusal/negotiation with respect to external 'triggers' (requests, commands, and the like).

Iteration of operations

This structure within an operations schema means that there is a checkpoint during the execution of the task (e.g. to check

properties of intermediate/external resources) and then a feedback necessary in the case of unsatisfactory result. It is possible to consider the productivity of loops and to evaluate their impacts on the basis of the involved operations: local operations, interactions, feedbacks.

Sequence of operations

Long sequences of operations should be carefully examined. Again, it is suggested to consider their productivity. They can be the result of an incomplete analysis which imposes a fictitious order and therefore the task could be better described by means of a concurrency structure. If, instead, they represent the actual nature of the task, then could be natural candidates for automation.

4. EVALUATION OF AGGREGATED OPERATIONS SCHEMATA

The analysis of operations schemata aggregated according to criteria apt to focus critical situations in the office, is fruitful to point out, for example: complexity and costs in terms of people, resources and facilities involved; executability of the office work, in terms of possible failure and blocking situations. Note that the parameters and the evaluations suggested for single operations schemata apply to the aggregated operations schemata as well.

All the tasks pertaining to (part of) an activity: procedure

This aggregation allows one to evaluate the classical aspects of the work flow. For example the following parameters can be considered:

- complexity, in relation to the number of involved roles, their operations, and the number and type of resources and facilities used;
- cost, derived from the previous parameters by assigning some value to time, resources and facilities;
- executability, in relation to the points of synchronization between the tasks (e.g., for communication purposes or for operations requiring the cooperation of different roles), which could give rise to total/partial deadlocks during the execution or to starvation (i.e., the indefinite postponement of the execution of a task w.r.t. others);
- volumes of produced and/or exchanged resources, execution times both of the operations and of the overall procedure, and the related ratios.

In addition, it is possible to characterize the occurrences of an operation within the procedure by considering: its pervasiveness, that is, if the operation belongs to several tasks; its repetitiveness, that is, how many times it occurs inside specific tasks; its relevance, that is, the amount of operations depending on its completion and/or outcomes. These parameters can be useful to evaluate if the operation is crucial and if its occurrences refer to the same operational conditions.

All the tasks pertaining to a role

This aggregation of operations schemata allows one to sketch the profile of the considered role, as the 'sum' of its characteristics inside each of its tasks, and the consequent work overhead.

The following parameters can be considered:

- the overall autonomy of the role for what concerns the handling of its tasks and the possibility of negotiation with respect to external 'triggers';
- the characterization of the role with respect to the control, coordination, consultancy, operative duties;
- the required skills both from the technological point of view and the managerial one;
- the degree of visibility of the overall productive processes in which the role participates;
- the degree of isolation/interaction with respect to other roles.

All the tasks of tightly coupled roles

This aggregation allows one to consider the mutual relationships between roles. Specifically, it is possible to evaluate parameters like:

- the degree of mutual acquaintance of the considered roles in terms of the visibility, compatibility and consistency of the expected mutual behaviours;
- the mutual hierarchical relationships and their coherence inside the organization.

5. CONCLUSIONS

Most of the parameters introduced in the paper refer to the structural characteristics of the net-based representation. Their evaluation takes advantage from the existence of a sound theory [Rei82] which provides well established criteria for analysing concurrent systems, and from the modularity (task-based composability) of the class of nets used in the OSSAD approach.

In particular, the invariants calculus defining properties of markings (S-invariants) or of set of repetitive actions (T-invariants) allows to consider the system dynamic behaviour and to identify, e.g., deadlocks or blocking situations, to study the sequences of markings to verify whether the system has a specific behaviour and to derive properties of the modeled system.

Another technique [Mil80], introduced inside net theory in [DP83], is the verification of the equivalence between the behaviour exhibited by two systems. Without entering into many details, this technique allows the designer to choose which operations are relevant for characterizing the system or system component behaviour and to verify if this behaviour is equivalent to another (predefined) one. This technique can be fruitfully applied in the evaluation of many parameters referring to roles: typically, mutual consistency, compatibility and coherence. In fact, the behaviour of a role interacting with another one has to be 'equivalent' to the one the latter role is going to expect from

the first and viceversa. Or, in the case of procedures, it is possible to verify if two different work organizations, e.g., with different workloads among the involved roles, provide to the rest of the system the same behaviour.

The quantitative parameters (like volumes, times, ratios) need the simulation mechanisms provided by the theory of Timed Petri Nets [Zub80] or Generalized Stochastic Petri Nets [ABC86], to study deterministic or probabilistic behaviours, respectively.

Finally, note that the existence of several automatic tools [PNN] supporting these analysis techniques, both on structural and simulation models, makes it realistic to apply them to (modular) real office systems representations.

ACKNOWLEDGMENT

The authors wish to thank their colleagues from the OSSAD (Office Support System Analysis and Design) ESPRIT Project (#285) for the encouragement of their work.

REFERENCES

[ABC86]

M. Ajmone Marsan, C. Balbo, G. Conte, Performance models of multiprocessor systems, MIT Press, 1986

[Con86]

D.W. Conrath (ed.), Manual: Office Support System Analysis and Design, CEE-ESPRIT OSSAD 285 Report, 1987

[DP83]

G. De Michelis, L. Pomello, A less restrictive Observational Equivalence notion, Proc. Fourth Workshop on Application and Theory of Petri Nets, Toulouse, 1983

[Mil80]

R. Milner, A Calculus for Communicating Systems, LNCS 92, Springer Verlag, 1980

[PNN]

Petri Nets Newsletter, Software Packages, 1985/1986

[Rei82]

W. Reisig, Petri nets: an introduction, Springer Verlag, 1982

[Sim87]

C. Simone, A language to speak about parameters for office systems evaluation, Proc. IEEE Workshop on Languages for Automation, Wien, 1987

[Zub80]

W.M. Zuberek, Timed Petri nets and preliminary performance evaluation, Proc. IEEE Symp. on Computer Architecture, La Baule (F), 1980

This research was developed with the financial contribution from IPACRI, Roma, in the framework of the ESPRIT-OSSAD Project #285.

Project No. 285

OSSAD METHODOLOGY: RESULTS OF THE ANALYSIS PHASE

Eduard Beslmüller,
IOT, Kästlenstr. 32, D - 8000 München 82

Sergio Caserta,
IPACRI, Via Rava, I - 00142 Roma

David W. Conrath,
Dept. of Management Sciences, University of Waterloo,
Waterloo, Ontario, Canada

Philippe Dumas,
Universite de Toulon, F - 83130 La Garde

In the framework of the project "Office support systems analysis and design (OSSAD)" concepts, instruments and procedures have been developed in order to describe and analyse office work. Field studies had been started in 1986 to improve the concepts and to validate the work done so far. The paper emphasizes the results concerning the descriptive model (the point of view of "how office work is done"). The concepts of this model are a good means of communication and support the isolation of weaknesses at the start of the analysis phase. Furthermore, since data collection instruments are associated with the concepts and their relationships, choosing the "right" ones and adapting them in the light of the weaknesses is possible in a short period of time. Therefore, during the whole analysis phase a systematic and efficient representation of weaknesses and a precise reconstruction of the actual practice is guaranteed. One specific strength of the Methodology is the fact, that it is not only directed towards technology. Performing the analysis it was appreciated by practitioners that also organizational and personnel aspects are taken into account. For example, it was possible to identify that in a certain office environment the through put time of a process could be improved by implementing technology, by rearranging competence levels or by reordering operations within tasks. Furthermore, the results are an enviable position to incorporate all three dimensions in the current work of designing alternatives.

1. INTRODUCTION

Concepts, models, instruments and procedures have been developed on behalf of the ESPRIT project entitled "Office Support Systems Analysis and Design." Although these were based on the experience of the researchers involved, we recognized that the only way we could be confident that they would work as intended would be to test them in the field. Thus, in 1986 we embarked upon three field studies, one each in France, Germany and Italy, to provide feedback that would either validate what we had proposed or suggest ways of improving the various aspects of our approach - the OSSAD Methodology.

... L N
- - - - -
- - - - -

The Methodology involves several phases: Contracting, Analysis, Design, Implementation and Audit. We now have experience on the conduct of the first two phases, and this paper presents the summarized results of this experience as it pertains to the concepts, instruments and procedures that were used. While these have been detailed elsewhere [1, 2, 3], we will provide a brief review so that one can understand our findings without having to refer elsewhere. Before doing this, however, we want to state several of the principles upon which the Methodology is based for they have had a major influence in the development of the approach. After the approach itself has been described, we will present our findings in terms of both the Methodology per se and some of the insights that we have gained about our subject organizations as a result of the analysis phase.

2. PRINCIPLES

Although we have not identified it as a principle, the most significant factor underlying the OSSAD Methodology is our insistence that the organizational (work flow and people relationships) and technical aspects of the system must be studied at the same time and as parts of an integrated whole. All contribute to the output of an organization, and to consider one (e.g. the computer-based support system) in isolation from the others can only lead to suboptimization. While most people seem to recognize that work content and flow and organizational structure (the relationships among organizational roles and the people fulfilling them) are related, and that work content and flow and the nature of the supporting technical system are related, they ignore the relationship between technology and structure. In the vast majority of cases new technology is introduced on the premise that it will support the existing organizational structure, knowing full well that work content and flow will have to change. The consequence is that a great many technological support systems fail to yield the anticipated benefits, either that or organizational structure is adapted to the new technology on an ad hoc and often costly (especially in human terms) basis.

Three principles which we have made explicit and deserve to be mentioned here are: Contingency, Iteration and Participation. The principle of *Contingency* reflects the observation that no one method is going to be appropriate for all office environments. As a consequence a Methodology has to provide a framework within which one can choose specific methods in response to the context in which they will be applied. To state it in another way, the methodology has to indicate an overall direction that is to be followed without fixing the details until one has some understanding of an office's environment. Furthermore, a methodology should provide a set of methods (instruments and procedures) from which one can select and adapt those needed for a particular situation (see Figure 1).

Iteration is an equally important principle. To put it succinctly, it is highly unlikely that any method or set of methods used for analysis, design and implementation is going to get everything right the first time. To find out what is right and what can be improved, which is the way we are approaching our own Methodology, we need feedback. Given appropriate mechanisms for feedback we can use

these to modify our analyses, designs and implementation procedures. But just one cycle cannot ensure that we are sufficiently close to our objectives to be confident that they have been realized. Hence, we have a belief in the need of iterative procedures.

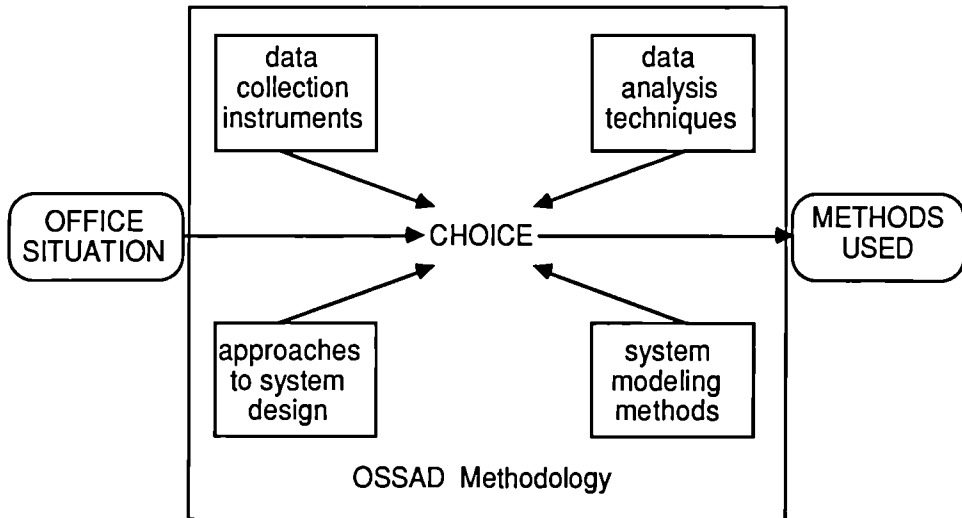


Figure 1: OSSAD Methodology

It is widely accepted [4] that user *Participation* is an essential element in the successful design and implementation of support systems. Without the active support of those involved in the system which is to be redesigned, it is most difficult to get either reliable and/or sufficiently complete data for analysis and design. Current users are better placed than others to provide insights regarding current problems and recommended solutions. Furthermore, even if the design activity could be done independently of its prospective users, without their participation one can expect system implementation to be both risky and costly.

While the above are stated as principles, both their applicability and the manner in which they are being applied are being subjected to field tests.

3. CONCEPTS, INSTRUMENTS AND PROCEDURES

3.1. Contracting

This first step in the OSSAD Methodology is aimed at the definition of the scope of the study, a preliminary understanding of the environment in which it is to take place, the identification of the objectives to be accomplished and the delineation of the instruments and procedures that are to be used. In addition, it is during the

contracting phase that the process of reciprocal education is begun - that of the "users" with respect to the OSSAD approach and terminology, and that of the "inquirers" in regard to the office's personnel, purposes and language. This is part of the process which we call "sensitization" - developing a basis for mutual understanding and realistic expectations.

There are two primary means for collecting data during the contracting phase. One is by going over existing documents and records, such as annual reports and job descriptions. These are likely to be limited in the insights they provide, but they are an inexpensive source of information. The other is by means of interviews with senior management and key personnel. The degree of structure of the interview depends upon the amount of information about the office's environment that can be obtained from other sources. The greater is the availability of such information, the less is the need for a structured interview, and the greater is the time that one can devote to "sensitization".

The output is a contract, whether formal or informal (i.e. not legally binding), that indicates what is to be done, when, how and by whom. This should include a statement of the objectives to be accomplished and an estimate of the expected costs in terms of both time and money.

3.2. Analysis

To obtain the necessary understanding, this phase involves both data collection and modeling. Since modeling provides the *raison d'etre* of data collection, the latter will be described in terms of the two categories of the models used during analysis. One we call the Abstract Model. It indicates what needs to be done to accomplish the goals and objectives of the office under study, and at the same time it defines the extensiveness of the analysis by circumscribing what is considered as relevant to the system. The other concerns Descriptive modeling. These models describe both the existing office and proposed alternative configurations. They indicate how the office accomplishes the objectives outlined in the Abstract Model. The relationship of the two models to each other and to those used for design is shown in Figure 2.

Each model can be described both verbally, by means of both a grammar and a glossary (where the terms are defined), and graphically. The ease with which this can be done and the effectiveness of these tools as vehicles of communication are also subjects of empirical verification.

3.2.1. Abstract Model

The Abstract Model is based on just two fundamental concepts. One concerns the division of an office/organization, usually based on the need to coordinate. At the highest level we refer to these divisions as *Functions*. These may be divided into Sub-Functions, and the finest division (the most detailed description of this hierarchy of work requirements) identifies *Activities*. The other concept considers the data/objects that pass between the Functions/Activities. These are grouped into what we identify as *Packets*. Thus on one hand we have the essentials of the work to be accomplished (Functions/Activities), and on the other the flow of

data and objects (Packets) required as inputs and produced as outputs. All of this, however, is in terms of what has to be done to satisfy organizational objectives rather than how the work is being (or is to be) done.

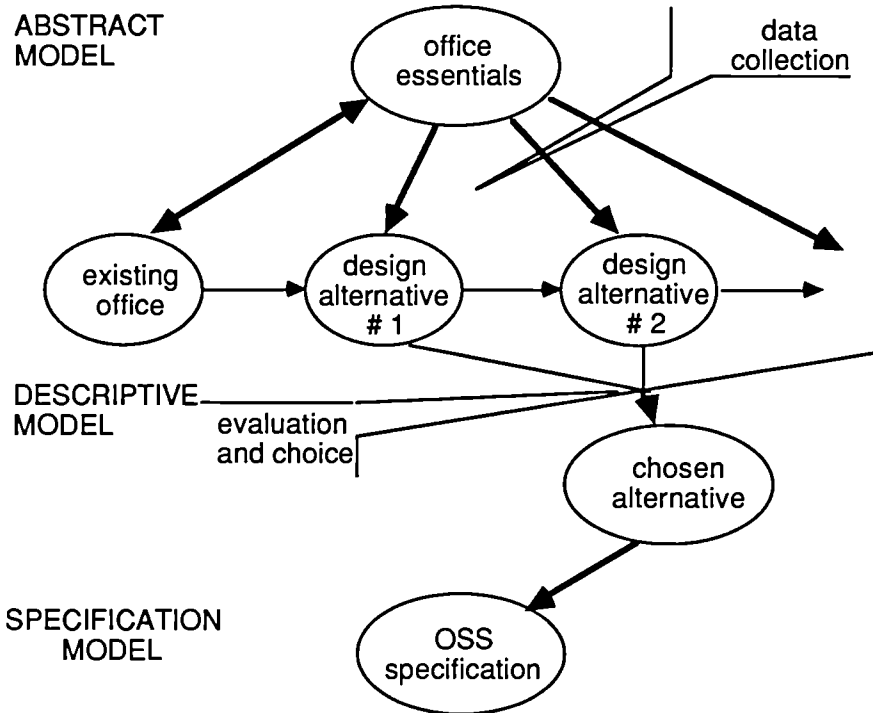


Figure 2: The Three Levels of Modeling

The data needed for the Abstract Model are collected by means of structured interviews using what we call the Office Functions Analysis form. Limitations of space preclude the inclusion of the form in this paper. Nevertheless it should be sufficient to mention that it contains a series of questions that move from the general to the specific, closing with questions concerning problem areas and suggestions of ways that these might be resolved. Objectives, critical success factors and criteria of performance are also elicited to provide the rationale for the Functions/Activities and Packets.

A key issue here is whether or not those being interviewed can think in terms of the abstract qualities required by the model in contrast to what they do in fact. Clearly the Abstract Model is of little use if it proves to be the same thing as the Descriptive Model of the current situation.

3.2.2. Descriptive Model

The Descriptive Model uses several building blocks (concepts). These are defined briefly as follows:

- *Operation*: the basic unit of work relevant to the description of an office.
- *Task*: the set of Operations which concern a given Role within an Activity (the intersection of Abstract and Descriptive modeling).
- *Procedure*: an aggregation of Tasks which may involve several Roles. It has definable inputs, outputs and measures of performance.
- *Role*: an organizational charge covering a set of Tasks performed by an individual or a set of individuals doing the same Tasks.
- *Unit*: an aggregation of Roles based on organizational requirements for coordination and control. Units may be aggregated into higher level Units, culminating at the entire organization.
- *Resource*: data/objects which are inputs to or outputs from Operations/Tasks/Roles/Units.
- *Facility*: Physical and/or technological support used to perform Tasks and Operations.
- *Actor*: an individual, with identifiable traits, abilities and attitudes, assigned to fulfill a Role.

From these we can construct: work content, organizational structure (inter-Role/interpersonal relationships and work flow), technical and/or physical support systems, communication networks and the set of human resources.

Because of the wide variety of data needed to build the Descriptive Model there is no one form of data collection that would suffice. Thus we have proposed several instruments and procedures. While existing documentation can provide certain information about Roles (e.g. job descriptions) and formal organizational structure, and questionnaires are useful for uncovering attitudes and personal preferences, and diaries and logs can detect the frequency and volume of communications and operations, most data collection revolves around structured interviews. Several forms that we use to aid the interviewer are mentioned.

The first is the Role/Activity Matrix. This is the means of defining the contents of a Task - that part of an Activity assigned to a single Role. Once Activities have been determined by means of the Abstract Model, and Roles have been identified (perhaps using an organization chart) the next step is the identification of Tasks.

After Tasks are delineated, we need to know how each is accomplished. For this purpose we developed the Task Description form. This is used to determine the various distinct Operations that comprise the Task, the inputs and Facilities used by each Operation, the output of each Operation, and measures of performance of each of these components where relevant.

Key issues here are the feasibility of collecting such detailed data, especially in terms of the level of effort that might be required, and the value of such data in enabling us to identify problems to be resolved and opportunities for improvement. In addition, there is the question of the ease of preparing the graphical representations of Operations, Tasks and Procedures.

4. FINDINGS ON METHODOLOGY

4.1. Principles

4.1.1. Contingency

This principle has been validated by the experiences during each of the field trials. In every case data collection instruments and procedures had to be adapted to local conditions. This was far more a consequence of organizational culture and climate than the differences in language and national culture, and thus we would expect this principle to hold for any methodology if it is to be successful across a variety of offices, no matter what the source of variety.

There was some concern expressed about the ease with which forms and procedures could be adapted. We feel that this can be improved both by more complete descriptions of forms and procedures and by examples of how these can be adapted to specific circumstances.

4.1.2. Iteration

Iteration proved to be extremely valuable from the point of view of the "inquirers". However, the users had a somewhat different perspective. Most didn't mind providing data as long as it was on a one-shot basis; but a number of people were less than eager to be interviewed periodically, especially when being asked to provide greater and greater detail. Several of them asked the question: is the extra detail really necessary? From the stand point of having a better understanding, one could say "yes". When the issue concerns the ability to provide improvements to the existing situation, an affirmative answer to the question of detail is far more problematic.

Our conclusion is that Iteration as a principle is valid. Nevertheless, it should not be used as a means of avoiding careful planning prior to the first data collection effort, nor should it be used without adequate justification with respect to the benefits to be obtained from further inquiries. Data providers are tolerant to the extent that data collectors appear to know what they are doing, and why.

4.1.3. Participation

We have no doubts about the value of this principle. On the other hand, putting it into practice is not always easy. In particular, it is often difficult to get middle and senior management to spend the time that we think is required for successful system redesign and implementation. Part of the problem was noted during our discussion of Iteration. The biggest issue is that they have day-to-day work which they have to attend to and which often has a higher priority than participating in a systems analysis and design study. This is the problem associated with almost all long range planning activity. Managers react to the immediacy of fighting fires rather than spending the time required to prevent them.

We have no new insights regarding this issue. Our primary suggestion is that a considerable amount of time and effort ought to be devoted to selling the importance of active participation by all

users and managers. They have to become convinced that without their participation the analysis and design team cannot be effective in solving their problems, nor in increasing their productivity nor that of their subordinates.

4.2. Contracting

The results of the contracting phase varied across the three field sites. At one extreme a formal contract was drawn and signed, which detailed what was to be done and by whom, including the expected level of effort and an associated budget. At the other was an informal agreement to cooperate, with a mutual understanding that things would evolve as the study proceeded and a recognition that both sides should be in agreement in terms of the nature of that evolution. In every case there appeared to be a common understanding of what would take place.

In general, the greater the emphasis on a formal contract, the less was the overall effort at "sensitizing" the organization; or to be more specific, the more the "sensitizing" was targeted at those who would sign the contract. At each field site we found that the process of reciprocal education was just begun during the contracting phase. The majority of the effort took place during data collection and analysis as "sensitization" is a continuous on-going process, and is essential to ensure cooperation.

Our conclusion is that the contracting phase is a necessary first step, and that it should involve more than just the formulating and signing of a contract. Mutual understanding of what is to take place, with the participation of whom and what objectives are being served appears to be an essential basis for long run cooperation. Furthermore, such understanding is not a one-shot affair. It requires continuous attention, and the burden rests on the shoulders of the OSSAD team.

4.3. Abstract Model

There are two general observations to be made regarding the development of the Abstract Model. One is that there was substantial variation in the abilities of various participants to grasp the meaning and the purpose of the Model. While at one site people embraced the concept almost immediately, even the term itself had to be changed at another. There was the perception that if it was abstract it was of no practical importance, and thus there was no reason to spend any time on it. The outcome was that a different term and vocabulary had to be used.

The other observation is that virtually every subject had difficulty distinguishing the concepts of the Abstract Model from descriptions of what actually takes place, even when they appeared to be able to distinguish the two notions. People can describe what it is they do, and they have the tendency to infer from this that what they do is what must be done. That is, the detailing of objectives and work essentials is likely to justify what is being done. Psychologically this is understandable. Nevertheless, we thought we had the means to overcome this predilection. We found, however, that large parts of the Abstract Models that were developed were based on inferences drawn by members of the OSSAD team rather than being derived directly from the data supplied by office system participants.

At a more specific level, the Office Functions Analysis form seemed to provide an effective interview structure. It was relatively easy to follow and to complete, though one should not ignore the problems identified in the preceding paragraph. Also, while the concept of Function could be grasped without great difficulty, our subjects had a much greater problem identifying Packets and their contents, particularly at an abstract level. They stated what was being communicated, not what was required as inputs or produced as outputs of Functions/Activities to satisfy organizational objectives.

The net result of all this is a certain dissatisfaction with the Abstract Model and its underlying concepts as they now stand. There is agreement that we need a conceptual framework that provides normative requirements or constraints for the Descriptive Models, especially those developed as new options. And thus we need something like the Abstract Model. The difficulties uncovered in the field in terms of its actual formulation, however, suggest that significant improvements ought to be made. Either we define and explain the existing framework more clearly, or we need to develop a better alternative.

4.4. Descriptive Model

The greatest part of our effort has been spent on understanding the current situation - the development of the Descriptive Model. We will discuss these experiences in terms of the concepts themselves, data collection instruments and procedures, and data representation.

4.4.1. Concepts

Most persons, both data collectors and data providers, had little difficulty with the concepts as defined in Section 3.2.2. As already noted there was some confusion between Packet (AM) and Resource (DM), though the bulk of the problem seems to reside in the Abstract Model. Also, upon occasion there was a confusion between Actor and Role - the person fulfilling a given Role. In essence, Roles became personalized. Since we have not yet made use of this distinction (its importance arises during system design), we cannot yet determine the significance of this problem.

Two expansions were suggested. One concerned the concept Unit. While there are no constraints regarding how Roles are aggregated into Units, most people see these aggregations in terms of organizational units as found in a typical organization chart. On the other hand, several participants wanted to group Roles/Actors on a different basis, such as common Tasks (e.g. financial accountants) or all the persons involved in executing a given Procedure (especially if employed by different organizational units and/or located in different places). The other suggestion came from the fact that conceptually we don't have an effective means for describing exceptions. The issue is whether this should be encompassed by a separate concept or by incorporating it into our verbal and graphical representations. We lean toward the latter solution.

4.4.2. Data Collection

Turning to data collection instruments and procedures, we have discovered that the Role/Activity Matrix was used at only one field site, and rarely there. The lack of its use suggests that it did not perform a useful service. In the few instances where it was applied, apparently subjects had difficulty identifying what part of an Activity was undertaken by a single Role. Furthermore, the Matrix was perceived as being somewhat irrelevant. Tasks could be determined more easily by examining Roles directly. Doing this, however, negates the one basis for ensuring an intersection between the Abstract and Descriptive Models. This is more evidence that we need to rethink Abstract modeling.

The Task Description form posed few problems for its users, though it did have to be adapted to the circumstances of each field site. It provided the basis for the Descriptive Models that were detailed, and was useful in that regard.

Each field site used other instruments and procedures as well. All found it necessary to develop the means to understand the flow of documents (that is, the information contained in them), and one site developed a specific instrument entitled Document Flow Analysis. At least at the field sites we encountered, the control of document flow was a major problem and required focused instruments and procedures. These will be added to our methods.

The use of tailored diaries and logs was also common. They served the purpose of gathering relatively objective data and provided one of the few means of quantification. Somewhat surprisingly, based on past experience, these were perceived as being easy to complete. Perhaps this was because they were designed to obtain very specific data, and thus were not as interruptive as the more general activity and communication diaries.

One other type of instrument was developed. The purpose was to identify problems, interruptions, delays, exceptions and the like. The only other form and procedure for identifying such phenomena was the application of the Office Functions Analysis. Even though this elicited a lot of information, it was collected during the formulation of the Abstract Model, and thus often failed to be included in the description of the actual situation. A separate form and procedure to be used during Descriptive modeling could be used to verify these earlier data and would be easier to integrate into the Descriptive Model itself since that would be its sole purpose.

4.4.3. Data Representation

Two issues arise here. One is the value of the models, especially their graphical representation. The other is the ease with which one can construct a model, given the existence of adequate data.

Regarding value, there is little dispute that the graphical models are an aid to communication and mutual understanding. The process of construction enables one to understand the underlying phenomena, and the use of the schemata to obtain feedback on their accuracy is of great benefit to both data collectors and data providers. There is less agreement about the verbal descriptions, and in particular

the careful use of the structured grammar. Most feel that it provides a level of precision that is unwarranted in the vast majority of circumstances. Since it is very laborous to use, the costs are often seen to outweigh the benefits.

The ease of model construction is inversely related to the complexity of the system being modeled. The consequence of this is the tendency to describe only simple systems, either by dividing complex systems into simple ones, or by simplifying them. In either case, the problems associated with complexity may be overlooked. The solution is to automate the model building process on a micro-computer. We recognize the inherent difficulties in doing this, though we hope to tackle the issue in the not too far distant future.

5. INSIGHT ABOUT FIELD SITES

As already mentioned all field sites expected improvements concerning Resources, e.g. letters, forms or cash. Although the focus was on different aspects of Resources, e.g. content, layout, structuring or making out of Resources, the overall goal was common to all sites, namely improving services given to clients. In what follows we comment on one investigation to state an example of the application of the Methodology (see also [5]).

While supporting Operations by means of Facilities or rearranging Operations within Tasks could yield improvements of office work an investigation of Procedures is aimed at figuring out requirements concerned with the aggregation of Tasks and their relationships with Roles. Figure 3 shows an (abstract) example of a Procedure and will be used for an explanation of the Document Flow Analysis (DOFA).

Several Actors are involved in this Procedure; they do not necessarily hold different Roles. The first one sends two documents one each to other Actors. Both of these try to accomplish their Tasks and in turn send their results to other Actors. The Procedure is initiated by the Task of Actor A and it is finished if Actor F has accomplished his Task.

The (non-adapted) instrument DOFA is aimed at the relation of only two Actors, the sender and the receiver. As data providers they are asked to state, e.g.:

- date of sending and receiving of documents,
- Tasks or Operations before and after a sending,
- documents transmitted,
- volume of documents (format, number of pages),
- kind of data (graphic, text etc), and
- weaknesses and inadequacies related to the items above.

The procedure -the manner in which data collection instruments are implemented- recommended for a DOFA is a docket. Both the sender and the receiver should deliver their data on one docket. This is to ensure a set of coherent cases for a future data analysis. In addition, it lowers the effort for data providers in giving, e.g. the documents.

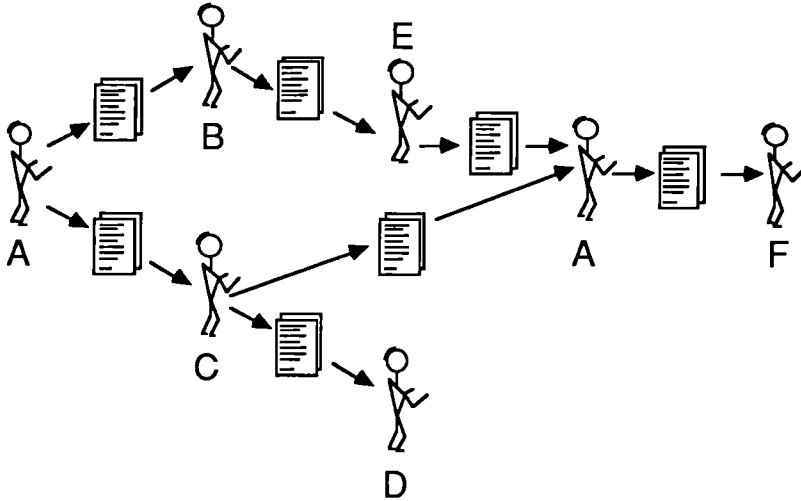


Figure 3: An (abstract) Procedure

Obviously, the DOFA has to be adapted in order to investigate Procedures. First, it has to be ensured that the dockets of one Procedure have an unambiguous marking with respect to this Procedure. This was easy to accomplish in the field site because there were only a few Roles who initiated Procedures. Second, the local conditions have to be taken into account. For example, some Operations could be predefined (on the dockets) because users participated in the adaptation process. Furthermore, some inquiries became no longer necessary because of irrelevance with respect to the characteristics of the field (e.g. no use of graphic).

One dimension of improving services is a faster reaction to a client's request. In the field site this meant a faster processing of Resources. From the stand point of a client the time interval from stating a request till an answering should be low. Taking the first and the last docket of a Procedure it is easy to compute this interval. A measure of productivity of the office is this interval too, but restricted to normal working hours. Both measures can be used for evaluation after an alternative office support system will have been implemented.

The computation of the through put time can be done by means of two dockets; an investigation of the overall processing time should be based on a path through a Procedure. (In figure 3 one path consists of the sequence A, B, E, A and F). Analysing the "productive" and the "unproductive" times (e.g. processing and transport times) can be also easily done. As a consequence of our analysis, the appropriate support of the transport of Resources was estimated to lower the actual through put time tremendously. It should be noted

that alternatives can be offered, either a technology is implemented or the internal transport service is done more frequently.

Finally we state three items related to all dockets within a Procedure. An indicator for the feasibility to lower the through put time is the ratio of the number of Actors and Roles involved within a Procedure. If this ratio is not close to 1 then one should investigate aspects of loops of proof-reading and of proof corrections but of course also the issue of control.

An indicator for specialization is the number of Roles involved in a Procedure. If this number is large then a likely consequence is a high division of labour which makes a lot of communication and coordination necessary. Taking into account the associated "unproductive" times it indicates a reason for a high through put time.

The last issue mentioned here is the ordering of Tasks within a Procedure. If most of the Procedures are accomplished in sequence then one should design a more parallel ordering to lower the through put time. Note, such a reorganization will require great effort and further data collection and analysis in order to guarantee a well organized parallel processing in Procedures. This is because it is likely that also Resources, Facilities, Tasks, Roles and even Units have to be changed.

6. CONCLUSIONS

The most obvious conclusion is that our work is far from being done. We are in the process of applying and "evaluating" the present state of the OSSAD Methodology in three separate field sites. From each of these come suggestions for changes in methods and approaches, and where consensus arises, changes will be made. Furthermore, the current work on the Design Methodology and its application in field sites will also influence concepts and procedures. Nevertheless we feel to follow the right path.

REFERENCES

- [1] Conrath, D.W., (ed.), Charbonnel, G., De Antonellis, V., Dumas, P., Simone, C. and Sorg, S., Manual Office Support Systems Analysis and Design (Universite d'Aix Marseille, Universite de Toulon, Institut für Organisationsforschung und Technologieanwendung, Istituto Per l'Automazione delle Casse di Risparmio and Università di Milano, Puyricard, La Garde, München, Roma, Milano, 1987).
- [2] Beslmüller, E., Conrath, D.W. and Simone, C., Bridging the Gap between Users and Vendors of Office Support Systems, in: The Commission of the European Communities (eds.), ESPRIT '85, Part 2 (North-Holland, Amsterdam, 1986) pp. 1025-1032.
- [3] De Antonellis, V., Bodem, H. and Coccia, A., Relevance of Models for Office Support Systems Analysis and Design, in: The Commission of the European Communities (eds.), ESPRIT '85, Part 2 (North-Holland, Amsterdam, 1986).

- [4] Bodem, H., Deinhard, C, Heinze, C., Matheja, E., Sorg, S. and Zangl, H., Feldprojekt Bürosystem, Akzeptanzbedingungen und Einsatzfelder eines integrierten Bürosystems mit Spracheingabe (Institut für Organisationsforschung und Technologieanwendung, München, 1986).
- [5] OSSAD-team, Comparative Evaluation of OSSAD's Analysis Phase, in preparation.

Project No. 813

PROTOTYPING AS AN AUTOMATIC TOOL FOR DESIGNING OFFICE SYSTEMS

Antoinette KIEBACK, Werner KERBER

DORNIER GmbH Friedrichshafen, Dept. TW 10, P.O. Box 1420
7990 Friedrichshafen, Germany

The project "Automatic Tools for Designing Office Systems" (TODOS) within the "European Strategic Programme for Information Technology" (ESPRIT), started in January 1986. Aim of the project is to develop tools to support office system design.

TODOS is divided in four Work Packages. This paper shows the Work Package dealing with prototyping within the project and describes the tool to be developed within this Work Package. "Rapid prototyping" designs a prototyping tool based on a conceptual model, designed by an other Work Package, developing basic office primitives, and defines a user friendly interface to the prototyping tool.

Prototyping is an effective alternative technique to traditional approaches for the development of information systems. A prototype is a model of the desired system, accurate in some ways but inaccurate in others. Users are able to relate what they see in the form of prototype directly to their requirements. The executable specifications strategy involves describing the software to be prototyped in a specification language which has some operational semantics.

This tool should have an executable prototype, based on AI-techniques at the end of 1987.
The relationship between TODOS and other projects like FAOR is presented.

1. INTRODUCTION

In this paper we will introduce in the idea of rapid prototyping especially for office information systems and there in the TODOS project.

The aim of ESPRIT Project No. 813, in area 4, Office Systems, "Tools for Designing Office Systems" (TODOS) is to develop tools to support office systems design. The tools will be used by the system designer and will support the phases of requirements collection and analysis, logical design, architecture design, and rapid office prototyping. The tools will consist of computer based instruments for collecting information, graphical interfaces for the proposed design models, consistency checking, and evaluation of the system being designed.

TODOS is a research project in a precompetitive phase. The main task of this project is the research of methodologies to automate information system de-

sign and the development of tools for practical proof of the investigated methodologies. Within TODOS rapid prototyping has a great part for evaluating and validating user requirements.

The design goals of a prototype are quite different from those of the final product. A prototype system is constructed to illustrate the feasibility of new ideas or of design. Unlike the final product, prototype design and construction often begins with incomplete specifications. Mistakes in its design are eliminated by quick and local modifications. Short cuts are taken to produce the prototype quickly at the expense of efficiency, robustness, generality and a good user interface.

We are developing a tool for rapid prototyping to show the user very early the implementation proposal of his requirements. As those requirements are modeled by the WP 2, we will develop a translator for these requirement specifications. Therefore, we have to analyse the TCM (TODOS Conceptual Model, developed by WP 2), to find a good language, supporting the basic concepts. There already exist languages and tools supporting rapid prototyping, a special tool will be developed to translate TCM automatically and perform a good user interface.

In the first section we introduce in the TODOS design loop, showing the dependencies in the project internally. Afterwards we show different proposals and aspects for realizing prototyping generally and specially for office information systems. For the approach of executable specification prototyping we describe the basic features of the TCM. In section 5 we study existing languages and tools for prototyping, matching basic TCM Concepts. The prototyping tool treating executable TSL will be introduced afterwards. The execution manager is a modul to control the user testing the prototype. The user interface will be described after the execution manager and shows the different views of the developer and the user. This paper will end with a short assessment of limits of prototyping.

2. THE TODOS DESIGN-LOOP

The design sequence for a computer-assisted office information system can be represented by a cycle. Starting from an actual OIS, the user requirements for the system under design are set up and analysed. These requirements are subdivided into functional and non-functional requirements. The functional requirements are integrated by means of a logical design into a conceptual model to describe the OIS. This model is the basis both for the development of an OIS prototype and for the design of a suitable OIS architecture, which also takes into account the non-functional requirements.

The OIS prototype created by rapid prototyping is tested for functionality by the user. Necessary modifications are integrated into the functional requirements and are analysed. This cycle is repeated until the prototype corresponds to the user's functional requirements. After that, design of a suitable OIS architecture is started which can lead to changes in the functional and non functional requirements when tested. Special design tools are developed for treating the individual work areas.

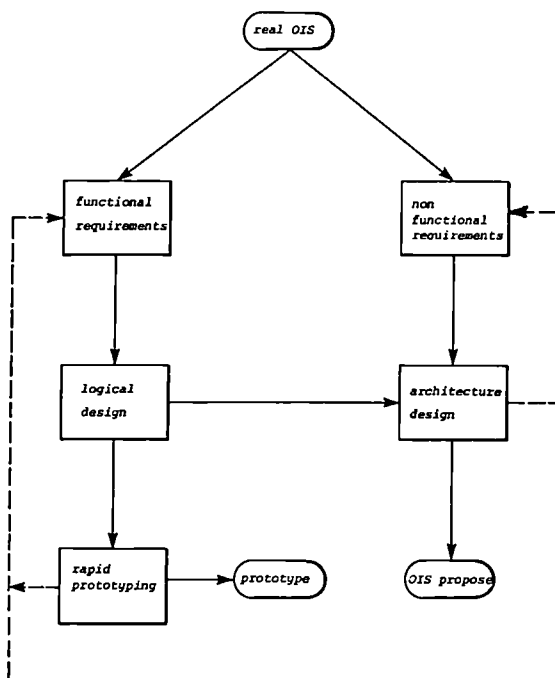


Fig. 1 TODOS Design-Loop

The field of requirements collection and analysis (WP 1), which is treated by the French company Sema-Metra and the Italian firm Italtel, serves to investigate user requirements for future office information systems. The results of these investigations then are evaluated with suitable tools in an analytic stage in order to obtain correct and complete specifications of user requirements. It is hoped that the most important requirements of the future office will be investigated in this way and at the same time the weak points of present OIS environments will be discovered.

The logical design (WP 2) of these OISs is elaborated in cooperation by the French software company Thomson Informatique Services, the Polytecnico di Milano, and the Informatics Institute Paris I University (Sorbonne). The objective of this work package is to create a development tool to design the functionalities of future OISs by means of the requirements analysis.

WP 3 is covered by Dornier GmbH. It is concerned with the development of a tool to support rapid prototyping of office information systems. The user has defined his own office environment based upon his personal requirements by means of the requirements analysis, which is represented by a logical model. On the basis of this model, the tool quickly generates a custom-made prototype for him. The functionalities of the OIS prototype are described by this model and can be provided by the user with the interface he desires. It is necessary to get a feedback from the user about his requirements at a very early stage.

The fourth work package (WP 4) is treated by the Dutch company OCE and the national Italian research centre IEI at Pisa (as subcontractor to Italtel). It deals with the development of tools for specifying computer System architectures suitable for the implementation of an Information System identified in the WP 1, and evaluating the performance of such architectures with respect to the workload representing the office activities.

The main objective of WP 4 [6] is to propose alternative architectures to implement a given office information system specification, and to provide the tools for evaluating such architectures from a performance viewpoint. To this end, a methodology and its supporting tools must be defined in WP 4 to produce, given a description of the data objects and the activities of an office information system, a specification of hardware and software components and of the connections among them that realize the functions required to the office information system.

Besides TODOS, the projects FAOR (Functional Analysis of Office Requirements) and OSSAD (Office Support Systems Analysis and Design) are concerned with related subjects.

The OSSAD project aims to create a model for an integrated office system, based on the most recent studies, but going as far as possible in a total integration.

The primary goal of the FAOR project is the investigation and analysis of office functions through the application of a new method being developed. This project will provide guidelines for the evaluation of available office systems, and will support the manufacturers choice of facilities to be integrated into the system.

FAOR could be connected to the TODOS project via WP 1. Both, FAOR and WP 1 are handling user requirements.

3. PROTOTYPING

The development process for OIS requires rapid prototyping, [5] evaluating and modifying the office system. Rapid prototyping is an effective communication link between developer and end-user, providing a medium for exploration of the end-user's needs. Integrated tools for evaluation and modification encourage this exploration, and are also valuable when updating applications which evolve over time.

As described in [7], prototyping techniques can be classified in three categories: scenarios, demonstration and versioning. A scenario will be expected to address interaction requirements and some functional requirements, although it cannot shed much light on application logic. A demonstration is likely to provide more insight into processing logic but may not be useful for evolutionary requirements, because the user's exposure is of necessity limited. A version 0 prototype is a working release of the system intended to receive use under conditions approaching the production environment.

Prototyping guaranties equivalence between specification and user requirements (verified by the user). Specification errors are not only identified at the end of the project, but in early software development stages. Corrections during those stages are more economic.

In order to be constructed faster than the final system, a prototype of a software project must ignore some aspects of the final system while preserving others. While a prototype need not be expected to embody all the assistance features of the final program product, the developer can note where more information is likely to be needed. Prototyping in TODOS is prototyping the functionality of the TCM and user interface. The prototype of our approach will be tested by a scenario, presenting the user an example of a system usage.

Rapid prototyping for OIS is a quick and dirty prototyping on testing specifications providing a fast feedback for prototyping. To meet the needs of rapid prototyping, we require that the development of a testable implementation from a specification be both correct and rapid. By the former we mean that the development process preserves the functional behaviour implied by the specification. To achieve this we use equivalence preserving transformations as our means of manipulating specifications into implementations, and rely upon machine support to perform the transformation applications. Good rapid prototyping tools are based on providing a development environment where what you see is what you get. This implies the use of screen-oriented tools where formatting can be accomplished with a screen editor and functions can be specified through menu selection. Products that require development through written programmes or by dialog with a long series of questions must be avoided. For this reason, tools are needed to produce prototypes of OIS for very early testing. This can be managed by executable specifications [9], which means testing the requirements by interpreting specifications.

In the logical design phase of TODOS a specification language has been developed (by Work Package 2), which will be interpreted for rapid prototyping. As seen before prototyping is no process, developing a final system at once. Prototyping defines a loop refining user requirements. A tool for prototyping in TODOS should therefore transform TCM in an executable version (a prototype), modify this version with respect to the modified TCM, corresponding to the user requirements and provide a user interface for the prototype.

Rapid prototyping is especially effective when implementing interactive office information systems. With right tools, the development process for these systems involves the generation subsystem which may be integrated into the total system.

4. SPECIFICATION LANGUAGE

The specification language TSL is described in detail in the Report 2.2 produced by WP 2 [4]. This overview is only meant to provide background information, needed for understanding the following sections.

The logical design phase of TODOS is based on a conceptual model for logical design of user requirements. These requirements are collected in the previous 'Requirements Collection and Analysis' phase of TODOS, and supported by computer based tools. The conceptual model is the basis of the following phases, the rapid office prototyping and the architecture design.

The TODOS Conceptual Model (TCM) integrates features from two models: the SOS Office Information System model [2] and the REMORA Information System model [3].

TCM has an object oriented approach, described in the TODOS Specification Language (TSL). The objects, used in this model were called entities described by their properties. Hierarchically the entities are structured using an 'IS-A' construction, corresponding a generalization of entities. For example 'secretary is-a person' means, that 'secretary' is less general than 'person'.

Two other abstraction constructs were used to describe the entities, the 'aggregation-of' and the 'association-of'.

'aggregation-of': divides the description of the entities in different properties.

'association-of': allows multiple property values.

The basic entities of the model are documents, objects, messages, agents, actions, and events. All persons interacting with the office information system are described by children of the agent entity. Actions are specifying the functionality of the system, therefore especially the entity action contains a property called steps, where function-calls are placed. These function-calls provide state changes. Actions are triggered by events and may handle messages, documents ... i.e. all entities. As an example we show basic entities (produced by Work Package 2) in the terminology of TSL (message, action), to demonstrate parts of our tool afterwards.

```

message:                                action:

<chief-decision> is-a message;          <application-creation> is-a action;
{aggregation-of                        {aggregation-of
  {envelope: aggregation-of            {values: aggregation-of
    {from: chief-clerk;                {comments: "...";
    to: SYSTEM};                       act-entity: application;
  follows: TRUE;                       steps: CR($a, $p);
  contents: aggregation-of              };
  {app: same-as application             in aggregation-of
    with {application-number, name};    {p: ref-to position_requested_letter};
  motivation: text;                    out: aggregation-of
  values: aggregation-of               {a: ref-to application}
  {description: {"the chief            }
    clerk enters his decision in      }
    the system through this message"} }
  }
}
}

```

These are features of the basic model, which will be blown up with information of the user requirements.

5. STUDY OF LANGUAGES AND TOOLS FOR PROTOTYPING

For deciding, which language and tool we can use for implementation, we have to analyse the conditions concerning realizing prototyping and transforming TCM.

As for prototyping, it is good to use languages and tools supporting

- changes, additions and testing of programs interactively, as prototype creation is a dynamic process,
- possibilities for programmatically creating other programs,
- comfortable and flexible system surfaces to provide user friendly interfaces, here for a designer and a test user,
- transparent system behaviour.

Concerning TCM, it is useful to have languages and tools, which have provisions for

- predefined hierarchical structure concerning the 'IS-A' structure of TCM,
- supports to model nested and recursive constructs with respect to the 'aggregation-of' and 'association-of' abstractions,
- an easy construction to model state changes provided by event/action concepts of TCM.

Studying languages and tools, classical programming languages like Cobol, PL/1 do not meet these conditions. Languages for system programming like C meet these conditions, but only at the expense of superfluous effort.

Using Artificial Intelligence-tools is an effective method for rapid prototyping. The language concepts, the design tools and many function modules of these tools allow realization of a flexible system with high functionality in a short time.

Prolog supports flexible database management with high level query facilities, but less comfortability in user interfaces.

LISP systems in conjunction with an object/flavor concept, as offered by LISP-machines (Symbolics, TI, XEROX etc.) are much more concerning the previous conditions, but not all of them comfortably. Some expert-system-shells running on a LISP machine, like KEE (Trademark of IntelliCorp), support a very powerful and comfortable programming environment, and allow automated programming.

For implementation of our prototyping tool and a prototype, AI techniques are used on an Explorer equipment with the KEE expert system shell in a Common LISP environment. The programming method of rapid prototyping allows a fast modelling of the problem solution. The main advantage is the quick setup of a prototype which already provides the restricted functionality of the final system without fixing the overall functionality. Thus the future user can get a relatively precise impression of the different system capabilities, especially of the user interface for "his" product, at a very early stage.

6. MODEL TREATED AS PROTOTYPE

For prototyping the user requirements, we are developing a prototyping tool, based on the TODOS Conceptual Model, which is described by the TSL and formally describes these requirements. This tool supports the methodologies of executable specification and partly automatic prototyping and allows the user to develop a user interface for his system.

The automatic part of the prototyping tool is the transformation of the TSL in an executable version. As described in the previous section, we used Common Lisp and KEE to develop the prototyping tool and the executable version of-TCM, the office prototype.

The TCM, modeled by entities, is reflecting the information of the user requirements, as described in the specification part. This information will be transferred from the logical design by a file containing the users requirements in the form of TSL, to the prototyping tool, which stores them after transformation in an executable version, in an office prototype knowledge-base. For changing requirements the tool has to modify this knowledge base. Therefore an other file will be transferred, containing the names of the deleted or modified entities.

The contents of these files will be transformed in the KEE-structure, which will not be described here [10].

Concerning the TCM, as in message and action, the entities are transformed into KEE-units with slots representing the entity structure.

Actions are described as seen before, storing informations about values and parameters, which are necessary for the "steps", where function-calls are placed. The functions (primitives), related to those calls are stored in a primitive library. This library can be updated, to allow a flexible use of new functions, especially concerning new technologies.

The non-automatic part contains the user interface generator, an editor to edit necessary functions, which are not part of the primitive library as seen before, and an instance generator, which adds instances to the entity classes provided by the TCM.

Instances are necessary to provide the user a test-environment with real data, which will initialize the office prototype.

7. FUNCTIONAL PROTOTYPING

Functional prototyping is computing the data transformation of the final system without necessarily using the final algorithm or the final command or display method. Often the computer, the language and the tools are different in the prototype from those used in the final system (but this is not necessary).

8. EXECUTION MANAGER

To test the office prototype the user will be supported by an execution manager, which controls the actions and events during execution time. Messages may be sent, which are related to events, which triggers actions also related to events and so on. All these state changes are controlled by this manager.

First, the user will log in the system to identify himself interacting with the system, that means the user has to be an instance of the agents unit. All action and event entities have authorization slots indicating which user is allowed to execute special actions, which has to be compared with the logged in user. Now the user is able to test the requirements he had specified, using the office prototype.

9. USER INTERFACES

In the following we will describe more deeply the shaping of the TODOS user interface from the point of view of Rapid Prototyping. Doing that we won't rather direct our attention to interface components as help or explanation components, user instruction or access control mechanisms, but we will point out the importance of the user interface with respect to Rapid Prototyping.

In the TODOS project, the user interface serves as an interface for the development of the prototype from the prototyping tool as well as access possibility of a future user of the prototype in forming his individual office environment on the screen. Because of this differentiation, we describe in the following two sub-chapters the user interface from sight of the designer as well as from this of the prototype user.

9.1. Developers View

From our point of view, the designer of the prototype should provide at any time of the functionality of the user interface, therefore all access possibilities of the tool like keyboard and mouse from the hardware side or command language and menu from the software part should be available. According to his requirements or the most efficient application, the user should decide unrestrainedly on the use of tool access.

By means of this user interface, the designer selects from the total functionality of the tool the desired functionality and arranges his intended prototype. When he acts then by means of commands, he operates on commands from

KEE and functions in Common LISP, choosing menus, he can select his functionality from the menus offered at the time.

For the time being the designer, interface looks like as follows:

- a Lisp Listener, on which the designer communicates with the tool by means of KEE commands and LISP functions,
- a HELP/EXIT menu, which offers help functions and explanation respectively allows to leave the tool,
- a window region to represent pop-up menus,
- a status area, where error messages, help texts etc. can be displayed,
- a work space to display file contents, to configurate forms or documents etc.

9.2 Users view

From users point of view, the user interface should look like as follows: each user must have the possibility to configurate his user interface at the beginning of his work with the prototyping tool or else at the start of the each work session. Just as the designer, the user can choose between the mouse and the menus or between the keyboard and the LISP Listener.

As the case may be, the screen can be described as follows:

- in the first case (mouse with menus), the screen looks like the user interface of the system designer, or the user arranges his screen from the offered (by default) possibilities.
- in the last case (keyboard and LISP Listener), the user acts on the tool by means of LISP functions or KEE commands. In a separate area, set down by the tool, a window displays messages or help texts etc.

Each user can save this interface in a sort of profile so that he can get his interface at each start of the tool or the prototype.

So the user of the prototype can use by means of the possibilities of rapid prototyping the advantages of this software technology in a twofold way: First, the user profits from the new possibilities of an easier development of the user interface by applying rapid prototyping techniques and secondly, perhaps the most important advantage of this technique, this so developed user interface lightens and improves the access to the tool and the prototype of TODOS.

10. LIMITS OF PROTOTYPING

Rapid prototyping is certainly a promising approach to this restricted problem area of designing an OIS tool. But it offers a comfortable frame to model very quickly user requirements in order to get an early feedback. This doesn't release the designer from filling suggestively this frame, it rests enough of effort in designing and programming in order to convert this concept to a marketable product. So we can neither overcome completely the immanent problems of rapid prototyping such as the transfer of software and tools, developed on huge and expensive AI machines, to middle-seized systems. Nevertheless rapid prototyping seems to be a promising attempt to manage the software crisis, to narrow the software gap.

11. CONCLUDING REMARKS

As TODOS is in a precompetitive phase, it is not necessary to have a unique hardware for the whole project. Therefore each Work Package researches methodologies to automate information system design for his special purpose.

As a result of the project, there will be a prototype for each Work Package, to visualize the usability of the researched methodologies.

The status of the Work Package concerning rapid prototyping is in implementation of the TCM-transformer of the prototyping tool and started designing the execution manager.

ACKNOWLEDGEMENTS

We thank all colleagues helping us to prepare this paper, in constructively criticizing previous versions and writing the desired form.

REFERENCES

- [1] : Lunghi, G. Business, Organization, Human Resources (ESPRIT TODOS-Internal-Report 1.1, Jan. 1987).
- [2] : Bracchi, G. and Pernici, B. SOS: A Conceptual Model for office Information Systems (Database, Vol. 15 No. 2, Winter 1984).
- [3] : Rolland, C. and Richard, C. The REMORA Methodology for Information Systems Design and Management (IFIP TCB Conference on Comparative Review of Information Systems Design Methodologies, North-Holland, Publishing Company, May 1982).
- [4] : Barbic, F., Fugini, M.G., Maiocchi, R., Pernici, B., Rames, J.R., and Rolland C. TODOS Conceptual Model and specification language (ESPRIT TODOS-Report 2.2, Jan. 1987).
- [5] : Kieback, A. Investigation of a prototyping model for office information systems (ESPRIT TODOS-Report 3.1 ISSUE B, March 1987).
- [6] : Heijmink, F., Meghini, C., Bledoege E., Dorsselaer, E. van, Aksit, M. and Musto, D. Architecture components analysis (ESPRIT TODOS-Report 4.1, March 1987).
- [7] : Carey, T.T. and Mason, R.E.A. Information System Prototyping: Techniques, Tools and Methodologies (INFOR Vol 21, No 3, August 1983).
- [8] : Barston, D. Rapid Prototyping, Automatic Programming and Experimental Sciences (ACM Sigsoft Software Engineering Notes Vol 7 No 5) pp. 33-34.
- [9] : Smaliar, S.W. Approaches to executable specifications (ACM Sigsoft Software Engineering Notes Vol 7 No 5) pp. 155-159.
- [10] : IntelliCorp KEE Software Development System User's Manual (1986).
- [11] : Steels, J.R., Guy, Common LISP (Digital Press 1984).

Project No. 234

COGNITIVE SIMULATOR FOR USER-INTERFACE DESIGN

P. Byerley⁺, P. Barnard^{*}, D. Carr⁺, A. Foster[°], T. Fowler[§], R. Saffin⁺, G. Ward[§]

The designer of human-computer interfaces needs to be able to predict human behaviour from interface design indices.

A feasibility study used multivariate techniques to relate a particular set of indices to human performance. Good descriptive fits were obtained for UNIX, Macintosh and VisiOn tasks.

Further analysis suggests that different types of interface may require different sets of indices and weights. Future work will require development of descriptive languages for designers, human-computer interaction models which indicate which aspect of the design should be measured, and techniques for processing descriptions to produce the indices such as expert systems.

Multivariate techniques offer considerable promise for error prediction.

1. INTRODUCTION

ESPRIT Project 234¹ had objectives which, in relation to Cognitive Psychology, were highly ambitious.

The key objectives were:

To test the feasibility of providing a software package that could be used by the human-computer interface designer as a design aid to assess human-machine cognitive compatibility. The prototype package is called the Cognitive Design Aid (CDA).

To provide paper-based design guidelines derived during the development of the design aid.

To provide an assessment of current trends in interface technology.

These ambitions were to be met in a two-year time span with a resource of 5.8 man years.

2. THE COGNITIVE DESIGN AID

The logistics of a quick feasibility study determined that the CDA should act on descriptions which are primarily of an interface, rather than of a task based, human-machine system.

⁺ Alcatel ESC, Harlow, UK; ^{*} MRC-APU, Cambridge, UK; [°] GEC Hirst Research Centre, Wembley, UK; [§] EASAMS, Camberley, UK

The indices are thus expressed in interface terms rather than attributes of cognitive processes².

The indices measure interfaces for two types of consistency (the consistency of the effects of actions, and the consistency of the causes for states), for the hierarchical structure of the system facilities, for the proportion of generic actions, for the human memory demands, and for the attention directing qualities of the interface screens.

The CDA uses a state-transition diagram technique for interface description; interface states are shown as nodes on the diagram, and actions are links between nodes which represent a transition from one state to another. The state-transition method allows the behaviour of the interface to be viewed as a whole, and is relatively simple for a designer to use.

The designer is required to specify a state-transition diagram, and then to augment this definition by describing types of state or act.

The 'types' are CDA user definable, thus providing flexibility. However, the users of the CDA have responsibility for defining their terms precisely and consistently.

The augmented state-transition definition is entered as a table. An example is given in Table 1.

TABLE 1: Example State-Transition Description

smdt-1.5

Project PROJECT 3 - MACINTOSH	Cognitive design aid
Facility EDIT A DOCUMENT	Session 21.1.87 2
Group EDIT A DOCUMENT	System Map Data Table
Element 1.5 FILE DOCUMENT	Page 1 of 1

Originating Node Number	Node Type	Originating Node Name	Next Node	Link Name	Link Number
>5 <	WINDOW	SENTENCE EDITED	13	GC-C TO FILE MN	5.1.13
6	WINDOW	DOCUMENT FILED			
11	FORM	SAVE AS SEL	33	TYPE NEW TITLE	11.1.33
13	WINDOW	C AT FILE MENU	14	GC-CLICK/HOLD	13.1.14
14	MENU	FILE MENU OPEN	32	PULL TO SAVE AS	14.1.32
32	MENU	SAVE AS HIGHL	11	GC-RELEASE BUT	32.1.11
33	FORM	NEW TITLE TYPED	34	C TO SAVE BOX	33.1.34
34	FORM	SAVE BOX HIGHL	6	GC-CLICK	34.1.6

3. VALIDATION STUDIES

The distinguishing feature of the approach which lies behind the development of the CDA is to assess cognitive compatibility by the use of a broadly based set of indices.

Consequently, validation of the CDA used the multiple regression technique.

Descriptions of the designs of already implemented interfaces were fed into the CDA, to provide the indices as independent variables.

Performance measures, principally errors, were taken from subjects who were required to execute a range of tasks using the interfaces. These provided the dependent variables.

Multiple regression analyses then indicated the weights by which the indices should be multiplied in order to predict the dependent variables as accurately as possible.

3.1. Phase 1 Validation³

The two systems chosen for Phase 1 assessment were the Apple Macintosh and UNIX, representing radically different design approaches.

Subjects were given a task sheet (Mac or UNIX as appropriate) requiring them to complete a series of tasks. The tasks were typical of those that might be required in a general office environment, e.g. creating, editing, and filing documents. All operator actions were recorded.

The major measure of human performance was error. An error was defined as a 'significant' departure from the correct and most economical action sequence to task completion.

The dependent variable for each task was average error per possible task action.

The results of the multiple regression analysis are presented in Table 2 and Figure 1.

TABLE 2: Regression Weights from Error Data For UNIX and Macintosh

Independent Variable	Coefficient
CONSTANT	20.435802
gnc	0.270781
ntc	-0.155719
ntac	-0.12263
mod1a	0.002331
mod2a	-0.29037
mod2b	-0.390261
mod2c	1.00773
mem1a	19.636085
mem1b	-39.906368

R-SQ. (ADJ.) = 0.8175 p = 0.0687

gnc	- the consistency of the effects of actions where nodes are not classified
ntc	- the consistency of the effects of actions where nodes are classified into types
ntac	- the consistency of the preconditions for nodes which are classified into types
mod1a	- the pyramidity of the interface facility structure
mod2a,b,c	- three alternative methods of calculating the proportion of generic commands
mem1a,b	- two alternative methods of calculating the proportion of occasions that the user is required to recall information

The attention indices were not included in the analysis

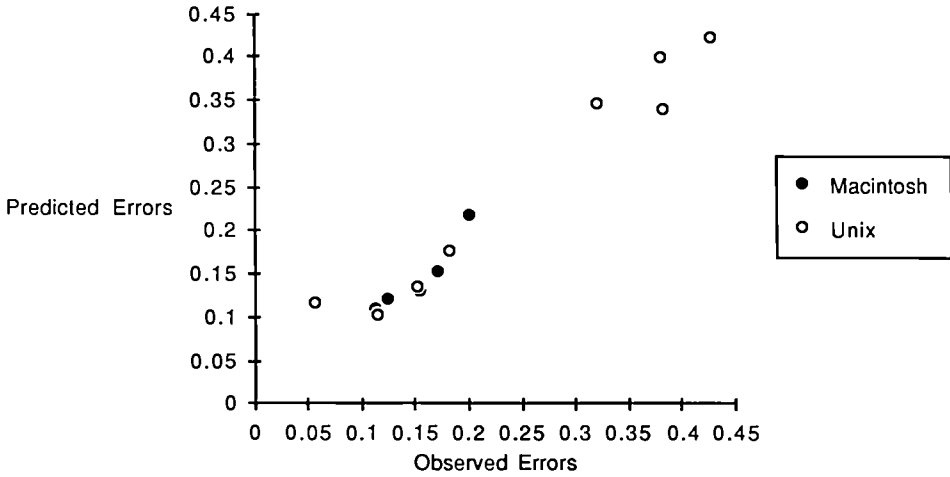


FIGURE 1
Plot of task observed errors against predicted errors for regression derived from Macintosh and UNIX data.

As can be seen a linear fit which is highly descriptive of the actual performance data is obtained from the CDA indices.

However, before rushing into a causal interpretation of the data some issues must be considered.

- i. The performance data were analysed from videotape by the same researcher who entered descriptions of the interfaces into the CDA.
- ii. The indices are highly intercorrelated, making causal analysis from the size of the regression weights very difficult.
- iii. The obtained regression weights are descriptive of this data only, another analysis is required in order to examine their predictive power.

Phase 2 of the validation was conducted in order to examine these issues.

3.2. Phase 2 Validation

Phase 1 validation employed the Macintosh and UNIX interfaces which are icon-based, and command language types respectively. Phase 2 validation examined VisiOn, which is lexical menu based.

For Phase 2 validation the user performance data had been collected for another research program. The interface descriptions for the CDA were developed and entered by a researcher who was unaware of the results of the VisiOn performance trials.

The results of the multiple regression are presented in Table 3 and Figure 2.

TABLE 3: Regression Weights from Error Data For UNIX, Macintosh and VisiOn

Independent Variable	Coefficient
CONSTANT	21.137068
gnc	0.050418
ntc	0.050037
ntac	0.016832
mod1a	-0.027524
mod2a	0.049214
mod2b	-0.214678
mod2c	0.179738
mem1a	20.541303
mem1b	-41.513812

R-SQ. (ADJ.) = 0.8514 p<0.0001

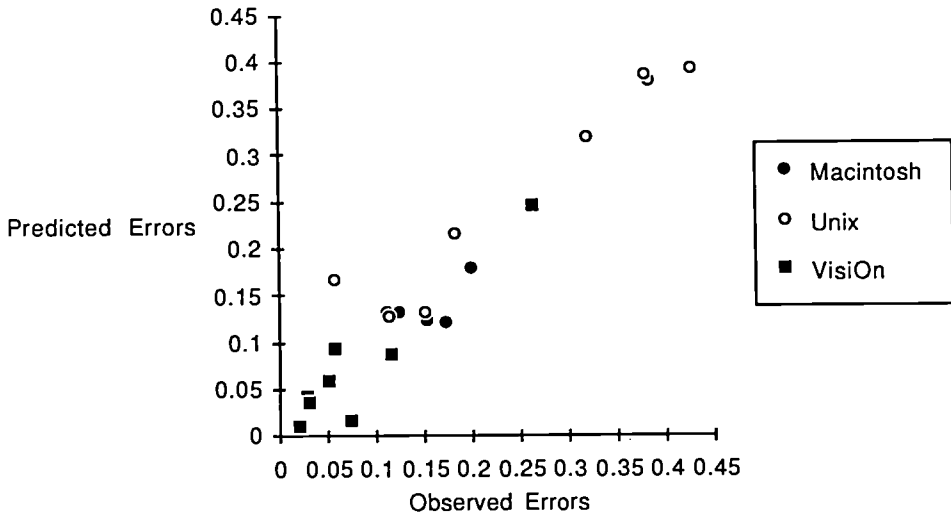


FIGURE 2
Plot of task observed errors against predicted errors for regression derived from macintosh, UNIX and VisiOn data.

Again it can be seen that a highly descriptive and plausible linear fit is obtained, for the clusters of UNIX, Macintosh, and VisiOn tasks. Furthermore, the patterns of weights shown in Tables 2 and 3 are quite similar. It was mentioned before that the intercorrelations of the independent variables prevents a precise causal model being induced from the indice weights. Some progress has been made in reducing the available set of indices to those representing distinct causal factors through the technique

of factor analysis.

Further progress will depend upon the work described in Section 4 of this paper.

If a tool such as the CDA is to be useful for the designer it must be possible to use weights derived from performance tests of one or more interfaces, on interface designs which were not used to produce the weights, i.e. the CDA should be able to be used predictively.

Thus regression weights were obtained from UNIX and Macintosh tasks and applied to VisiOn CDA indices. The resulting predicted task errors correlated with observed errors. ($r=+0.76$, $p<0.03$) This result is shown in Figure 3.

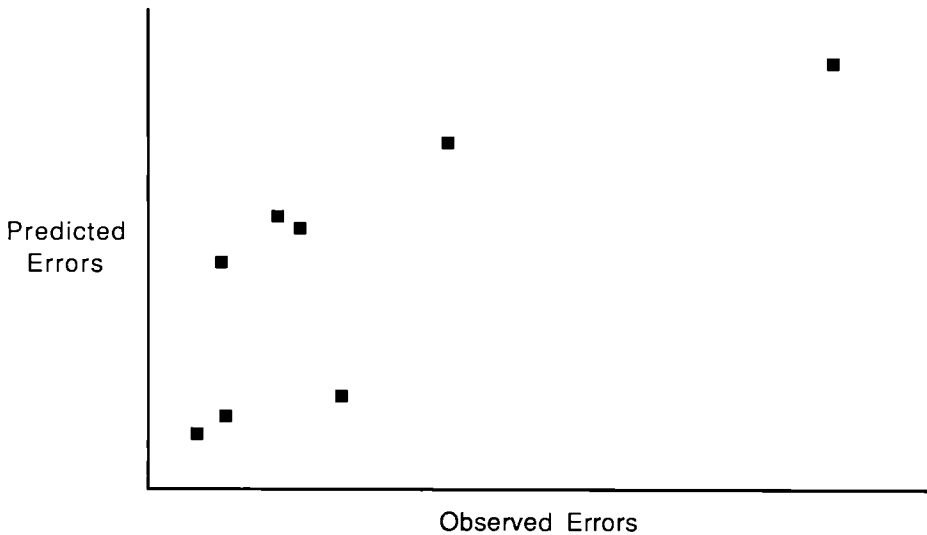


FIGURE 3
Plot of predicted errors against task observed errors
for VisiOn, using regression derived from Macintosh and UNIX data

It could therefore be argued that there might be a causal model common to UNIX, Macintosh, and VisiOn tasks.

An alternative viewpoint could be proposed. Interfaces might be classifiable into types. Each type could require a different causal model and thus a different subset of indices with appropriate weights.

A designer would then be required to categorise his/her interface design in order to select the appropriate cognitive evaluation.

Some evidence for this can be obtained from a more sophisticated analysis.

A regression model was obtained from VisiOn, and off-sets were obtained for the constant and the weights which would give the best fits for UNIX and Macintosh. None of the off-sets for Macintosh were significant, indicating that the VisiOn model could be applied to Macintosh. However, one of the off-sets for UNIX was significant (for a consistency index). Plausibly, Macintosh and VisiOn could be categorised together and contrasted to UNIX. These results are highly tentative since only a subset of the indices could be used.

3.3. Conclusion

The findings obtained from the validation work support the multivariate method of which the CDA is an early example. That method is to describe and predict user behaviour with an interface on the basis of a number of indices. Complex multifunctional systems determine human behaviour in a multicausal manner. The multivariate approach can reveal multiple interacting causes in contrast to the univariate approaches which have tended to predominate in this field.

The next stage in this research is the provision of more sophisticated indices, and user performance testing to reveal the present range of canonical interfaces with their respective causal models.

This requires three types of analysis, (i) an analysis of the language used to produce descriptions of designs, (ii) an analysis of what part of the design the descriptive language should be applied to, (iii) an analysis of the processing that should be applied to the design descriptions.

4. FUTURE WORK

4.1. The Descriptive Language

The technique used for the description should satisfy two equally important requirements. Firstly, because the resulting description will be automatically evaluated the technique must allow precise, rigorous descriptions. These descriptions will consist of two parts:

- i. The creation of a problem language. This contains data types for denoting human, machine and interface entities and their attributes, and a vocabulary for system or user actions which affect the values of the attributes of the entities. Since user actions are often dependent on perceiving and organising information from a number of sources the descriptive technique should allow the denotation of organised information e.g. the screen state. Also, since the users' actions can alter this state, the technique should allow easy denotation of state-to-state transformations. Most tasks require sequences of actions, it should, therefore, be possible to represent sequences. Finally, the denotation should reveal similarities between actions, and between entities, allowing the statement of generalities.
- ii. Axioms which model the behaviour of the interface or more comprehensively the man-machine system. These axioms provide the meaning of the actions.

Secondly, the technique should have specification utility i.e. it should be compatible with the way humans reason, particularly about interfaces (at the simplest level this means that it should not lead to descriptions which are very difficult to read), and the resulting descriptions should be relevant and complete for a Human Factors analysis.

There is not yet such a technique.

Although the state-transition approach is proposed to be relatively simple to use it suffers from a number of problems.

- i. It can only be used to represent state-transitions which are caused by discrete actions. It cannot, for example, be used to represent a computer game with continuous input where the number of states/actions is infinite.
- ii. It cannot be used to represent an adaptive interface, e.g. one for which the state-transition specification would change as a consequence of an action.
- iii. It cannot be used to represent higher level patterns or generalities about an interface.
- iv. The description cannot incorporate a dialogue history since the descriptive entities are merely states and actions, and states cannot incorporate descriptions of previous actions.
- v. Additional formal descriptive techniques are required to specify actions and states.

Future development of a CDA must we think require a descriptive technique, usable by designers, which is much more powerful than the state-transition approach. This issue is made even more salient by the following discussion.

4.2. The Application of a Descriptive Technique

Following the selection of a descriptive technique, to what part of the design should it be applied?

Description of the interface itself can provide highly useful input to a Human Factors analysis. This is demonstrated with the Cognitive Design Aid for the error scores.

Further Human Factors analyses could be undertaken by extending the application of the initial description. Let us consider the types of questions that could be asked from the results of the description of the interface only.

- How many levels of decomposition of the interface are possible?
- Do the commands always have the same effect regardless of context?
- How many different commands are there?
- Are interface states always the result of particular commands?
- Are there indicators to attract the user's attention to information necessary for the execution of a command?
- etc.

Questions that cannot be asked from the interface only descriptions are such as

- How many ways are there of achieving a particular user goal?
- Do the available actions match the most typical user goals?
- How similar are the action sequences for a set of related goals?
- On how many occasions is a cognitive process of that kind required?
- Does the interface supply clear information about the past, present and future of goal achievement?

- How adequate are the help messages?

Examples of approaches which could be trowled to provide indices which can be defined in more cognitive terms than the CDA interface indices are Polson and Kieras [1], and Payne and Green [2], with their Task Action Grammar, and Barnard [4].

4.3. The Processing of the Design Descriptions

The CDA applies simple algorithms for calculating its indices. It is quite likely, however, that analysis of descriptions of the human-machine system requires processing for which algorithms are unavailable.

In this case, the appropriate technique is the application of heuristics and theoretically derived approximate rules within an expert system. An example of an expert system in this area is that of Barnard et. al. [3], which is derived from the Interacting Cognitive Subsystems Theory proposed by Barnard [4].

FOOTNOTES

- 1 Project 234 Consortium: Alcatel ESC, Harlow, UK (Prime Contractor); MRC-APU, Cambridge, UK; GEC Hirst Research Centre, Wembley, UK; Logos Progetti, Milan, Italy.
- 2 B. Christie and M.M. Gardiner were responsible for the specification of the indices.
- 3 Much of the data for this study was collected by R. Wicksman.
- 4 This work is fully reported in 'User Learning of Core Command Sequences in a Menu System'. M.D. Wilson, P.J. Barnard, and A. MacLean, September 1985. IBM Hursley Human Factors Laboratory, Report HF114.

REFERENCES

- [1] Polson, P. and Kieras, D., A quantitative model of the learning and performance of text editing knowledge, in: proceedings of the CHI '85 ACM conference on Human Computer Interaction, San Francisco (1984).
- [2] Payne, S. and Green, T., Task Action Grammars: a model of the mental representation of task languages, Human Computer Interaction (1986), in print.
- [3] Barnard, P., Wilson, M. and MacLean, A., Approximate Modelling of Cognitive Activity: towards an expert system design aid, in: Carroll, J. and Tanner, P., (eds.), CHI and GI: Human Factors in Computing Systems and Graphics Interfaces (Toronto, 5-9 April 1987) ACM New York pp. 21-26.
- [4] Barnard, P., Interacting Cognitive Subsystems: A psycholinguistic approach to short term memory, in: Ellis, A., (ed.), Progress in the Psychology of Language (Lawrence Erlbaum, London, 1985) Vol. 2, Ch. 6. pp. 197-258.

Project No. 878

THE ProMInanD DESIGN CYCLE AND EVALUATION PLAN

Knud MØLLENBACH and Anette B. PETERSEN

Section for Information Processing and Cognitive Studies, SICOS
Risø National Laboratory, P.O.Box 49
DK - 4000 Roskilde, Denmark *

ABSTRACT: *ProMInanD* stands for *Extended Office Process Migration with Interactive Panel Displays* and is an integrated approach to the design of office systems. The approach is based on a concept of office worker influenced migration of office processes as well as novel interactive panel displays. Thus the project will result in two novel office technologies: a *Migration Server (MS)* and an *Interactive Panel Display (IPD)*. These are briefly presented. A supportive *human factors analysis and evaluation* to optimize office worker satisfaction are also carried out. To that end the project has given it a high priority to relate the functionality of the two technologies to a solid conceptual understanding of the office. A framework is suggested which allows for a broad spectrum of perspectives including anthropological, social, and organisational perspectives. It is shown how the framework can be used to study the design cycle and develop an evaluation plan in relation to the development of the two office technologies. In conclusion the paper explains how the project was led to test of the technologies in carefully chosen real world applications.

1. INTRODUCTION

The full title of ESPRIT project 878 is *Extended Office Process Migration with Interactive Panel Displays*, its acronym being *ProMInanD*. The project presents an integrated approach to the design of office systems. The approach is based on a concept of office worker influenced migration of office processes as well as novel panel displays. The effort is combined with a supportive human factors analysis and evaluation to optimise user satisfaction. The project consortium has three partners - from Germany IABG mbH, and from Denmark Modulex A/S and Risø National Laboratory.

The objectives of ProMInanD are *the exploration of:*

- The proper combination of automated migration of office processes between office workstations and the somewhat non-deterministic behavior of persons in office roles.*
- New possibilities offered through the development and application of a prototype model of a novel office panel display device.*
- A prototype system into which extended office process migration and office planning panels are integrated.*
- Human behavior in dealing with migrating processes as well as an investigation of the acceptance of the prototype panel display device by office workers.*

The integrated information system aimed at in the first part of the listed objectives is called a Migration Server (MS). It will support offices where processes are performed which consist of steps to be carried out in parallel or sequentially by persons in different office roles, especially if the number of steps are not known in advance, if there is a need for deviations from predefined procedures, or if non-deterministic human behaviour is to be taken into account. The MS

* Present and permanent address for both the authors:
SCAITECH A/S, Scandinavian AI Technologies,
Krogbjerggård, Havegærdet 1, DK - 2800 Lyngby, Denmark.

operates with the help of a system of stations connected by a communication medium. Activities migrate to the stations concerned. A station is a workstation - e.g. an office computer on an office worker's desk - or a service station - e.g. a usual mainframe offering its services [1].

The migration of a process is automatically performed according to a formal description. A process may be at the same time at all those stations where steps can be carried out. By calling process operations an office worker can affect his work on a step or the migration of a process. He can interrupt the work at a step, e.g. by putting it on a pile, by inserting another process, by postponing it for renewed submission when a certain date has arrived or information concerning that process is received, or by back checking with a preceding office worker. He can influence the migration e.g. by making a selection among the incoming processes, by assigning priorities, by withdrawing, by terminating, or by simple forwarding. Support for different steps is achieved by the possibility of integrating proper programmes which are automatically selected by the migration server for execution.

The Interactive Panel Display (IPD) is a novel device developed on the basis of the well-known MODULEX [Note1] planning boards. It is computer-controlled and consists of a set of building blocks comprising intelligent display and feedback elements. At any time, the elements can be manually placed on and removed from a modular board of arbitrary size. The panel display can be used for the entry of a range of different kinds of plans or their modifications into a computer as well as for the representation of computer-generated or controlled plans. A simple example is the assignment of persons to office roles which, in turn, could affect a process migration. Furthermore, the panel display can be used as a tool for the design, specification, and simulation of office processes and, in general, for planning jobs [1].

It is important that the research and development of the project can be seen as an original contribution to the office systems area. Throughout the project it has been considered important that the work is performed with reference to a solid conceptual framework to ensure compatibility of the project results with other developments in the office systems area. Also the development of the novel technologies is complemented by a parallel effort on the development of a methodology for their introduction with proper regards to human factors, and to ergonomic, organisational and social aspects.

Rather than giving the detailed description of the technologies, we concentrate on the latter subjects.

2. A MULTI-DISCIPLINARY FRAMEWORK FOR STUDY OF THE DESIGN AND EVALUATION OF THE TWO TECHNOLOGIES.

The analysis of complex systems has been dealt with from numerous viewpoints and using various methods based on very different theoretical approaches. Systems theory, control theory, and computer science have contributed considerably to the understanding of technical systems. However, for the studies of office systems it has also been necessary to include knowledge from social science such as anthropology, psychology, management theory, and organisation theory.

Intuitively most people know what an office is, but even a simple discussion shows that the term is conceived in many different ways. In fact one could call the term *an office* a semantically unstable term. Several authors have discussed the subject and proposed a variety of definitions. Examples can be found in Newman [2] who puts the emphasis on the office as an information processing unit, in L.Aiello, D.Nardi and M. Panti [3] who stress complex and dynamical interaction of people, or in R.Hirschheim [4] who underlines the social environment with social interaction. We shall not make an attempt to formulate yet another definition of the office concept, but stress that much office work includes decision making.

Of course the situation calls for care when the objectives of a project are to develop, interrelate, and evaluate two novel office technologies. Thus it has been given a high priority to relate the findings and results of the project if not to a clear definition of the office, then to a common conceptual understanding of the office. We know that the development of a thorough understanding of the office is a major objective for other ESPRIT projects, e.g. project 56, FAOR[5], and the project 285, OSSAD [5], [6], but results from these projects have only recently

become available. To make up for this lack of background information we developed a conceptual framework to ensure that project 878 had a common terminology defined in relation to a common frame of reference and to make it not only possible but rather straight forward to relate our results to the results from other ESPRIT projects.

The framework has been designed top-down and the detailed development has only been made at very high abstraction levels. As will be clear below, the framework has served us well in several tasks:

- to set up general requirements for the technologies,
- to limit our search space in study and choice of methodologies,
- to understand similarities and differences in the design cycles of the two technologies,
- to design the evaluation of the technologies,
- to overview the potential applications of the two technologies.

Two points have been considered very important in the development of the framework. All the time one should be able to have the *richest possible picture* of the office and the framework should promote a *holistic view* of the office. The thorough review of office views and perspectives by R.Hirshheim [4], the soft systems methodology by P.Checkland [7], the logical structure and basic methodology in the cognitive task analysis by J.Rasmussen [8] and the discussions by J.Rasmussen [9] have served as the basis for our approach. A detailed discussion hereof is found in the project's first deliverable, [1] and in the report from the ESPRIT workshop at Schäffergården[10].

As the office is a bad starting point for a description of the framework we will start by considering an *organisation* of which the *office* can be conceived as a part. An organisation is a collection of *humans*. A *context* holds them together. At their disposal they have some *technology*. This defines the boundaries of the organisation. Under the heading *human factors* we will consider characteristics specific to the individuals' relations to the context and the technology, while other characteristics, e.g. the interaction of the individuals and the relations between groups on one side, and the context or the technology on the other side, are given the heading *human actors*. The organisation will of course interact with its *environment*, but this will not be discussed further.

The conceptual system	Real world	Model world: Schools
Human actors	<ul style="list-style-type: none"> - People - Groups - Organisations - Social entities 	<ul style="list-style-type: none"> - Anthropology - Organisational thory - Social psychology
Human factors	<ul style="list-style-type: none"> - Individuals 	<ul style="list-style-type: none"> - Cognitive psychology
Context	<ul style="list-style-type: none"> - Work 	<ul style="list-style-type: none"> - Systems theory
Technology	<ul style="list-style-type: none"> - Equipment 	<ul style="list-style-type: none"> - Computer theory

Figure 1. The Conceptual System.

An organisation can be studied either in the *real world* or in the *model world*. This latter term is equivalent to the systems world used by P.Checkland[7]. Figure 1 shows the chart spanned by these two dimensions. The entities found in the real world and in the various domains of description are shown in the left column and the right column contains the top level of the model world approach in terms of the headlines of the theoretical schools.

Along other dimensions we could describe various specific models and analytical methodologies, but in this paper the chart shown will serve our purpose.

3. THE DESIGN CYCLE.

Let us first use the framework for a simple case -the analysis of the ProMinanD design cycle. The Technical Annex of the project [11] has a detailed description of what is called a *workplan*. But this is a design cycle and a simple one. The very fact that the project is formulated and realised stems from a *problem recognition* leading to formulations of *ideas*. The starting point for the development of the new technologies is a phase with *requirement specifications*. Once they are established a *design specification* can take place. This is the basis for an *implementation* phase where after *operation* can start. When some operation experience is gained the requirements can be adjusted and a new loop in the design cycle starts. Some of these phases take place in the *real world* and some in the *model world*. This is indicated in figure 2.

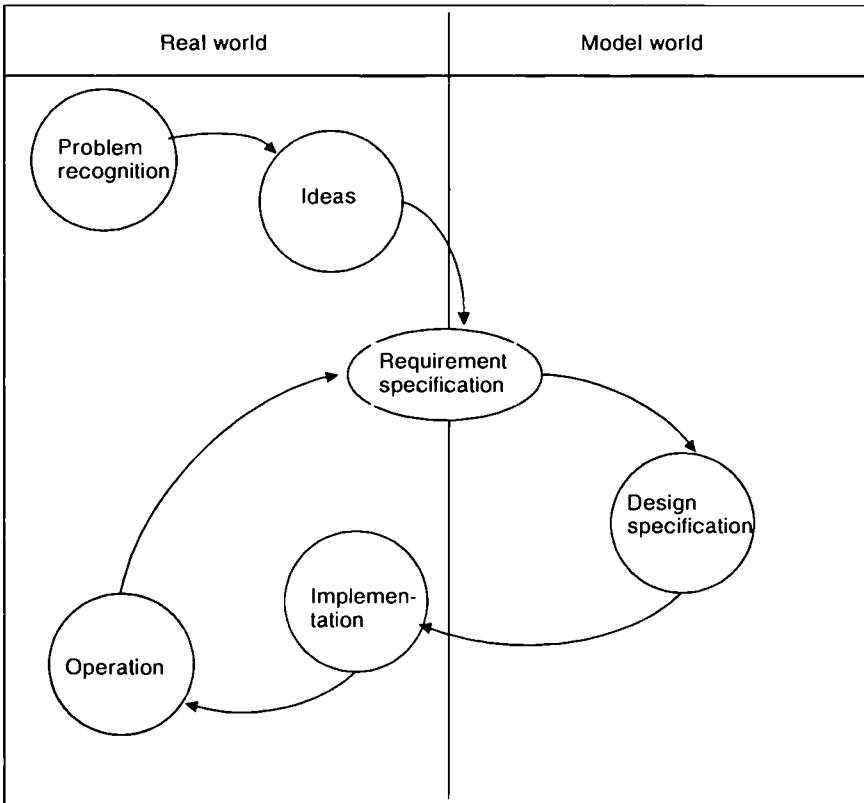
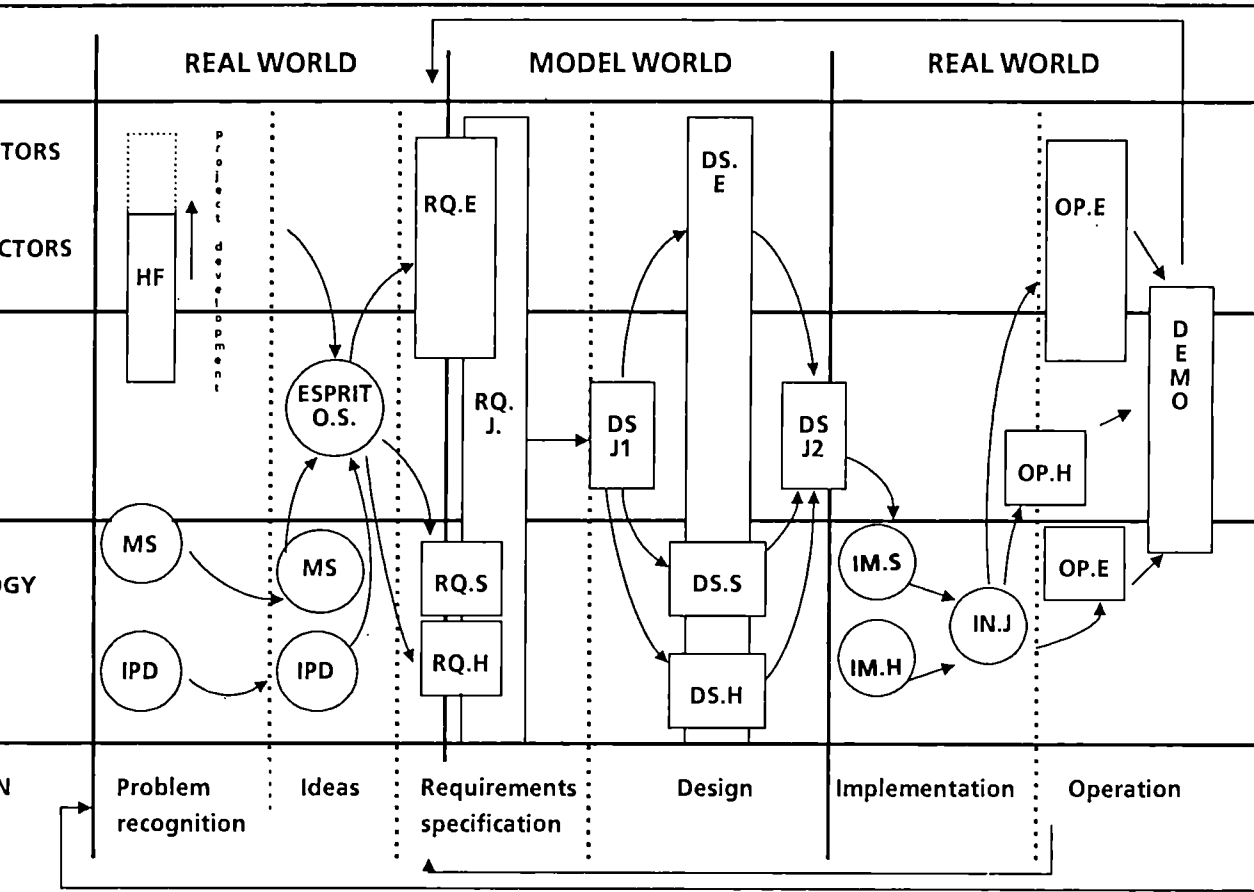


Figure 2 Simplified Design Cycle.



Complex Design Cycle

With the prime dimension of the framework in mind a more sophisticated analysis of the design cycle can be made. This analysis will make it clear in what phases who contributes to the various aspects of the design. Figure 3 is an attempt to make this analysis of the projects design cycle. Note that tasks have been subdivided into S, H, E and J components. S, H, E and J are short for Software, Hardware, Ergonomics and Joint respectively and indicates both the center of gravity for the subtasks and the responsible partner. Software is the main part of the MS, hardware for the IPD and ergonomics is a part of the human factors and evaluation effort.

Figure 3 clearly shows that it indeed is possible to develop novel technologies without considering the context or the people they have to interact with. However, we strongly believe that such an approach has severe shortcomings. They will be clear from the chapters below.

4. EVALUATION.

In the project the evaluation is supposed to be an integrated part of the design process. The purpose and strategy for an evaluation of new technologies as a part of a design process should be approached quite differently from the evaluation needed during and after the implementation of mature technologies.

The purpose of the evaluation is to make sure that the requirements are fulfilled, and also to optimise office workers jobsatisfaction, and to combine the technological development with a supportive human factors analysis.

As we design *Third Wave Office Systems*[12] technologies to be used in the office in the 1990's, we must make sure that the MS and the IPD also live up to the general assumptions for these systems. There will be a change in focus of interest as new office systems emerge: The systems of 1980's focus on *form, individuals, products, and window systems*, whereas the systems of the 1990's are concerned with *contents, teams, process, and information viewers*[13].

In spite of the complex mass of methods and literature inside the area of systems evaluation in general, and office systems in particular [14], [15], [4], there are no clear methods for evaluation. Numerous platitudes have been offered about what should be done. Most of the literature covers specific parts of the evaluation - evaluation of the system implementation which focuses on user satisfaction [16], efficiency, effectiveness [14] or cost-benefit [17], or the evaluations directed toward technical verification of the technology, in accordance with questionnaires based on requirement specifications for the technology and general requirement recommendations [18]. Others emphasise the human-computer interface [18], [19], [20], [21], [22], [23], but only a few pay attention to the evaluation of impact on social and organisational aspects of the technology [24], [25], [26], [27], [28].

In our framework of integrated office systems, we propose an holistic approach covering a variety of office perspectives, to get a richer picture of an office[1], [5]. By *holistic* we mean that the framework can include all the above mentioned perspectives and methodologies and show relations among them.

It is our intention to demonstrate how the framework can be used present as an evaluation plan, covering different evaluation strategies in relation to the design of the MS and the IPD. Figure 4 summarises the discussion, the purpose of the evaluation in the project and illustrates the evaluation concept of the project. It may be seen as an overall guideline for the evaluation plan in the project. For the development of the evaluation plan, the evaluation concept has been related to the design cycle. It illustrates how the evaluation effort and the design hang together.

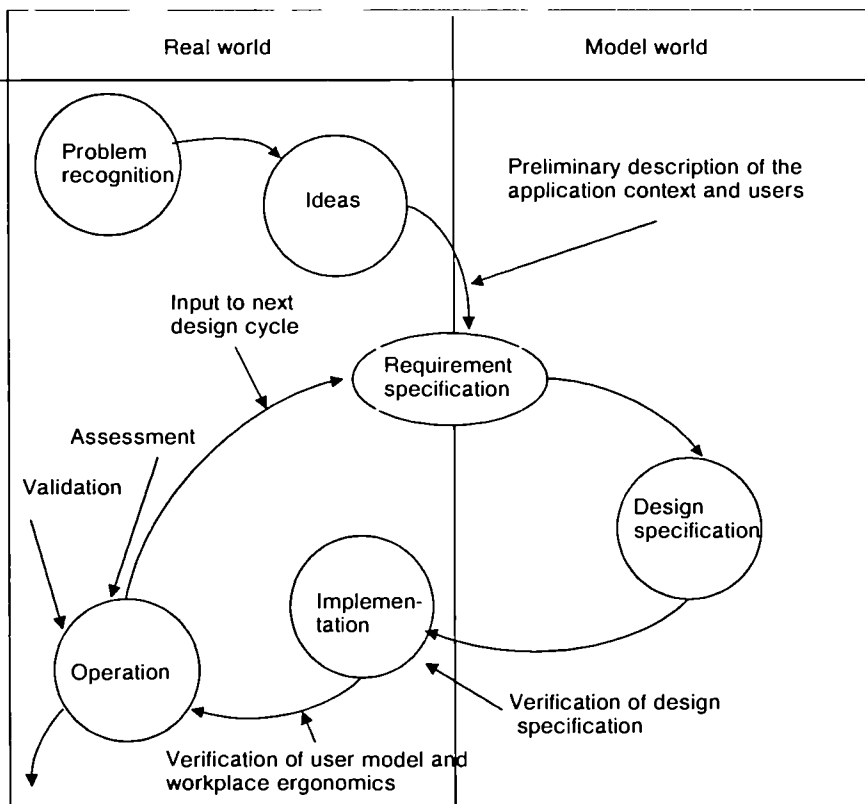


Figure 4 The evaluation cycle as an integrated part of the design process.

4.1 AN EVALUATION TAXONOMY.

The kind of evaluation which has always been practised in one form or another is a *technical verification*. The technical verification considers whether the design is correct or in other words, do the results satisfy the requirement specification. A technical verification is based on a system engineering approach and is normally practised by the designers themselves. D.Gertman [18] has developed guidelines for technical verification, with the purpose of offering "a ready-to-use and easily updated framework for applying *human factors engineering guidelines* to involving interactive computer based design or systems one is considering purchasing and implementing".

Considerations concerning the question *is it the right design?* - are not approached in the technical verification. This perspective on evaluation is normally called *validation*. The validation requires user's involvement: hence, it is of course impossible for designers objectively to answer the question *is it the right design?*. It is important to let the users evaluate how the system functions, and whether the system objectives are realised.

For technologies under development, this part is often done in a laboratory, e.g International Computers Limited has set up a laboratory equipped like an office. Users are as research subjects invited to work in a laboratory with the technology together with their normal work [29]. Similar projects are found in other computer manufacturing companies and the design process of control room.

As we conceive the validation, it also includes *work place ergonomics*. Thus, the interface between the users and the technologies may be considered.

It is important to consider the new features offered by the MS and the IPD when developing the evaluation plan. The IPD can be used for support of group processes. How these features influence the work, the organisation, and the social structures is a perspective which must be emphasised during the evaluation. The MS will support each office worker in fulfilling her task, and it will bring the tasks direct to the office worker whom it must concern. It will to a high degree influence the office worker's possibilities to change her own job. These kinds of support will of course also influence the work, the organisation, and the social structures.

Neither the verification nor the validation is supposed to include evaluation from an organisational, social, or anthropological perspective. These evaluation perspectives have largely been ignored. An evaluation including these perspectives requires an evaluation of the technologies in an application driven function in a real world environment. This kind of test is not normal during a design process, but is rather what had been done to some degree when implementing systems for use. We believe that an evaluation plan including this perspective will result in new requirements for technologies and give some good experience for the implementation of the System.

In focus	The technology	... and the user	... in a context	...in the real world
Approach	Engineering	Work place ergonomics	Human factors	"Holistic systems approach"
Evaluation taxonomy	<u>Verification of design specification</u>	<u>Verification of work place ergonomic requirements</u>	<u>Validation of user model and human-comp. interaction</u>	<u>Include assessment of social and organizational impact</u>

Figure 5. Evaluation Taxonomy.

4.2 THE EVALUATION PLAN.

The following specifies the evaluation of the IPD and the MS. After the assessment of the integrated use of the IPD and MS, (point 7 in table 1), the first loop in the design cycle will be completed and the *Basic Panel System* (BPS) developed. According to the projects time schedule the BPS is available by July 1988. Point 8 and 9 are related to the second loop in the design cycle, and after that, the *Extended Panel System* (XPS) should be available. That effort completes the work in ProMInanD.

1. The general evaluation plan for ProMnanD [30]
2. Verification of the IPD.
3. Validation of the user model and the interaction between office workers and the IPD in a real world application labeled CC [Note 2]
4. Assessment of a first version IPD in a real world application labeled CC
5. Verification of the MS, based on the design specification.
6. Validation of the user model and the interaction between office workers and the MS in the real world application, labeled MAA [Note 3].
7. Assessment of an integrated use of the MS and IPD based on results from the application labeled CC.
8. Evaluation of MS in the application labeled MAA
9. Evaluation of an integrated use of the MS and IPD in the application labeled MAA.

Tabel 1 The evaluation plan.

4.2.1 VERIFICATION.

The verification of the IPD shows whether the IPD satisfies the requirement specification. The objectives under consideration are retrieved from the requirement specification [1]. It should be kept in mind that a panel without electronic support has been manufactured for several years as a planning system. Many of the basic features are well proved and will not be tested. The verification of the IPD will concentrate on the new electronic features.

The verification of the MS is closely related to features concerning the MS and not the technologies on which it is based. The guidelines developed by D.Gertman [18].offer a consolidated checklist for verification of technologies in accordance with the design specification. In relation to our framework the checklist only offers a single perspective on the technical verification. The guidelines are developed when menu driven user interfaces were a common standard. Today workstations offer window- and icon-based interfaces. The MS is developed for SUN Workstations [Note 4] with UNIX environment [Note 5] and they offer a modern user interface. It implicates that it is necessary to reformulate and expand the checklist. Thus the checklist will be used as a first step verification and as inspiration for a more project-specific checklist for the technical verification.

4.2.2 VALIDATION

A validation will, as mentioned above, require users' involvement, because it should answer the question -is it the right design? It could be application oriented users or it could be test persons using the systems for different specified activities, with the purpose of testing predefined features, e.g ergonomics, and interface.

The user model is important and should be evaluated both during the validation and, if possible, also during the assessment in a real world situation. It should be validated whether the designer user model is in accordance with the user's conception of the technology.

During the project the importance of having a *common application* was emphasised. The methods for designing, implementing, and evaluating office systems have undergone considerable development during the 1980's. When the project proposal was written two years ago, evaluation - verification and validation as specified above - of new technologies was done by designers and possibly by users in a technical environment. For ProMnanD two common applications have been chosen: *container tracking* at the C.Clausen Lineragencies Ltd.[Note 2], and *apron worker control* at an airport, at Munich Airport Authorities [Note 3]. The two applications will be used both for validation purposes and for assessment purposes.

4.2.3 ASSESSMENT

The assessment is split up in two steps. The first is based on the *Critical Incident Method* and the users are allowed to explain their impressions of the technologies without any influence from the research in form of questions. The second step will be based on the users' experience and,

further, it will focus on questions such as, job satisfaction, influence on the social structure, and influence on control relations. Let us comment on some of the latter subjects. *Will the use of the MS and IPD influence job satisfaction?* Today this is a key question when implementing new technologies. Job satisfaction has been the issue of a number of studies [16]. Thorsrud & Emery [31] has proposed a number of psychological job demands used in several studies. The possibility of *contentment, challenge* and *variation* in the job has been stressed as very important when introducing new tools to support office workers. For a long time the introduction of new technologies has often had a negative influence on contentment, challenge, and variation in the job often with routine work, no personal development, and no possibility of making decisions concerning one's own job activities as a result [1].

Will the introduction of the MS and the IPD result in a change of social relations? If the social contact and human support are changed or even eliminated, important information relevant to the task may be lacking. Another influence is the lack of social relations concerning personal development through different kinds of information from other office workers. Automated office process migration may result in negatively influenced social relations. As one task should be delivered from one office worker to the next, the office workers will communicate and transact. Only a part of this is directly or strictly relevant to the actual work migration. Other parts could relate to dissemination of values - as a support for the corporate culture - or might give a more detailed context for an activity. To some office workers the company irrelevant chat is essential for their motivation and job satisfaction, and the few wasted minutes can be a very profitable investment from an efficiency point of view

The human communication might well be important sources for knowledge on the work context and on the general relevance of the job. Use of automated work migration will perhaps call for a compensating effort - be it more frequent and informal department meetings, or extra coffee breaks. Thus, introduction of automated work migration might show some side-effects.

Automation of work migration gives rise to change in activities as the migration system is implemented and will not be a simple duplication of existing manual migration processes. As the tasks change, the roles of the office workers by whom they are performed will change too. In other words, the control relations will change. Of course these changes can well be coupled to questions on job satisfaction. But perhaps of equal importance is the coupling to the organisation. The introduction of an MS will probably be followed by needs for some organisational adjustments. For both the MS and the IPD a series of security questions arise and must be clearly defined and answered.

On the other side, new relations could be established if the IPD is used by groups of office workers. As a multiuser tool the IPD may prove to support new social relations. It is well established that *group problem solving* from an innovative and creative viewpoint can be advantageous. The direct manipulation of a common tool with a common overview and possibilities of a fast response to all kinds of suggestions seems to promise useful support for a group social system.

5. CONCLUSION.

The experience from the ProManD project has shown us how important it is to be able to relate all details of an office technology design process to a solid conceptual understanding of the office.

The main reason is as follows: all office technologies can be studied from a variety of perspectives and viewpoints and an internal coherency can only be established if they all can be related to a common frame of reference. This holds not only for the mature technologies but also for novel technologies and the design process through which they see the light of day.

Even though establishment of a conceptual understanding of the office is not a major objective for this project we have suggested - in a sketchy form - a possible structure for a common multidisciplinary framework for the study of offices and people and technology related to them. We believe our framework after further developments is a good candidate for a common framework in which results from other ESPRIT projects can be linked.

In particular we must point out that our framework lends support to the existence of a holistic approach to office systems, and thus, it is possible to use a social and an organisational perspective along side with the more commonly used technology driven perspective.

ProMInanD has attempted a design integrated evaluation. It demands a stringent evaluation plan. To that end we have developed an evaluation taxonomy and on top of that developed a detailed evaluation plan.

This plan spans from technical verification in one end to assessment of technology acceptability in the other. We have chosen to test the technologies in modest but real world applications. We are sure this work will lead to a number of new requirements and an experience that will prove invaluable in successful implementations of the two technologies. If so ProMInanD has contributed to the IT industry with a design strategy which to a much larger extent than existing ones can be called a *user- and context driven design*.

ACKNOWLEDGEMENTS.

The ProMInanD project team is very well working and the authors wish to express their gratefulness to our friends and colleagues at IABG and MODULEX for their never failing enthusiasm in the projects technical discussions and in relation to our work.

We would also like to thank the SICOS group at Risø for interesting discussions, assistance and advice from the very start of this project.

The inspiring guidance from and collaboration with professors Niels Bjørn-Andersen and Jens Rasmussen is gratefully acknowledged.

Our review experts professors Mathias Jarke and Panos Constantopoulos, Dr. Ron Easterby and Mr. Ralf Hansen of Commission of the European Community are thanked for their deep interest in ProMInanD.

REFERENCES AND NOTES

- [1] Møllenbach, K., Petersen, A., Ramsperger, N., Andersen, M., Bischeltfrieder, H., Karbe, B. and Weimann, P. Office worker Interface of the Basic System (First Deliverable ESPRIT project 878. 1986)
- [2] Newman, W., Office Models and office systems Design, in: Naffah, N. (ed.), Integrated Office Systems - Burotics (North-Holland, IFIP 1980)
- [3] Aiello, L., Nardi, D., Panti, M., Modelling the office structure: A First Step Towards the Office Expert System, in: Ellis C. (ed.), ACM-SIGOA Conference on Office Information Systems (SIGOA v5n 1 & 2, 1984)
- [4] Hirschheim, R., Office Automation: A Social and Organizational Perspective, (John Wiley and Sons. 1986)
- [5] Commission of the European Communities, Information Technology and Telecommunications Task Force, ESPRIT Project Synopses, Office Systems (1986)
- [6] Conrath, D.W., Antonellis V. De., Simone, C. An Approach to Modelling Office Organization and Support Technology, in: Bjørn-Andersen, N (ed.), Modelling the Nature of the Office for Design of Third Wave Office Systems (ESPRIT Workshop, Scæffergaarden, Denmark 1986)

- [7] Checkland, P., *Systems Thinking, Systems Practice*. (Wiley, 1981)
- [8] Rasmussen, J., *On Information Processing and Human Machine Interaction. An Approach to Cognitive Engineering*. (Elsevier, 1987)
- [9] Rasmussen, J., *A Framework for Cognitive Task Analysis in Systems Design*, (Risø-M 2519, 1985)
- [10] Møllenbach, K., Petersen, A., *Terms and Concepts for Modelling of Office Systems*, in: Bjørn-Andersen, N (ed.), *Modelling the Nature of the Office for Design of Third Wave Office Systems* (ESPRIT Workshop, Scæffergaarden, Denmark 1986)
- [11] ProMInanD, ESPRIT project 878, Technical Annex, project document (1985)
- [12] Bjørn-Andersen, N (ed.), *Modelling the Nature of the Office for Design of Third Wave Office Systems* (ESPRIT Workshop, Scæffergaarden, Denmark 1986)
- [13] Halvorsen, P.K. of XEROX PARC, Personal communication, (1986)
- [14] Bikson, T.K. and Eveland, J.D., *New Office Technology: Planning for People. Work in America Institute*, (Plenum Press, New York 1985)
- [15] Hamilton, S. and Chervany, N.L., *Evaluating Information Systems effectiveness - Part I: Comparing Evaluation Approaches*, (MIS Quaterly, September 1981)
- [16] Mumford, E., *Values Technology and Work*, (Marinus Nijhoff, 1981)
- [17] Schäfer, G., *Benefit Analysis of Office Systems -Concepts for a Method* (Proceeding at Information Systems Assessment IFIP WG 8.2 Working Conference. 27 -29 August 1986, The Netherlands)
- [18] Gertman, D., *Aiding the Design of Information Systems: Human Engineering Checklist for Interactive Systems*, (3rd Symposium on empirical Foundations of Information and Software, 1985)
- [19] Verplank, W.L., *Designing Graphical User Interfaces*, (Proceedings of CHI'1986 Human Factors in Computing Systems Boston, Massachusetts, April 1986)
- [20] Verplank, W.L., *Results of STAR Icon Tests (Part 1 and Part 2)*, (XEROX, OPD Systems Development Functional Analysis Group July 1979 and March 1980)
- [21] Rouse, W.B., *Computer - Generated Display System Guidelines, Vol. 2: Developing an Evaluation Plan*, (EPRI Electric Power Research Institute, Prepared by Search technologies Inc., Norcross, Georgia, September 1984)
- [22] Rouse, W.B., Frey, P.R. and Rouse, S.H., *Classification and Evaluation of Decision Aids for Nuclear Power Plant Operators*, (Oak Ridge National Laboratory Report No. 8303 - 1, March 1984)
- [23] Rouse, W.B., *Designing and Evaluation of Computer-Based Decision Support Systems*, in: Salvendy, G. (ed.), *Human-Computer Interaction*, (Elsevier Science Publisher, 1984)
- [24] Rubinstein, R. and Hersh, H., *The Human Factor, Designing Computer Systems for People*, (Digital Equipment Corporation, 1984)
- [25] Hirschheim, R. and Smithson, S., *A Critical analysis of Informatin Systems Evaluation*, (Proceeding at Information Systems Assessment IFIP WG 8.2 Working Conference. 27 -29 August 1986, The Netherlands)

- [26] Robey, D., Farrow, D.L. and Franz, C.R., Assessing Conflict in System Design, (Proceeding at Information Systems Assessment IFIP WG 8.2 Working Conference. 27 - 29 August 1986, The Netherlands)
- [27] Kling, R., Social Analyses of Computing: Theoretical Perspectives in recent Empirical Research, (Computing Surveys, Vol. 12, No. 1, March 1980)
- [28] Kling, R. and Iacono, S., Computing as an Occasion for Social Control, (Journal of Social Issues, Vol. 40, No. 3, 1984)
- [29] Otley, S. of ICL (International Computers Limited) Personal communication, (1986)
- [30] Petersen, A., Evaluation for ProMinanD, ESPRIT project 878, (Technical Note, 1987)
- [31] Thorsrud, E., and Emery, F., Mod Nye samarbejdsformer, (Steen Hasselbachs Forlag, 1970)

[Note 1] Modulex is a trademark of Modulex A/S.

[Note 2] The application labeled CC is an application at C.Clausen Lineragencies Ltd using the IPD to support container tracking. Details are at this time not publicly available.

[Note 3] The application labeled MAA is an application at Munich Airport using the MS and the IPD to support apron service on aircrafts. Details are at this time not publicly available.

[Note 4] Sun workstations is a trademark of Sun Microsystems, Inc.

[Note 5] Unix is a trademark of AT & T.

Project No. 901

ASPECTS OF MULTI-MEDIA RETRIEVAL SHOWN BY ESPRIT 901

HILL Patrick

YOUNG Nigel

SOCIETE EUROPEENNE DE PROPULSION

52, Quai de Dion-Bouton

92800 PUTEAUX

FRANCE

LOGICA

64, Newman Street

LONDON W1A 4SE

ENGLAND

1. INTRODUCTION

Project ESPRIT 901 is titled 'An intelligent public data, voice and picture storage retrieval system'. The project uses the new Laservision ROM (LV-ROM) technology developed by PHILIPS to demonstrate effective methods of retrieving large volumes of data, voice and moving and still pictures. This paper describes the various retrieval employed in the demonstrators for accessing the different media.

The first part of project 901 is the development of the DOMESDAY project demonstrator. DOMESDAY uses two LV-ROM videodiscs to store approximately 1 Gigabyte of data, 100 000 still photographs and 30 moving film sequences with sound. The database is accessed using indexed keyword retrieval based on a semi-expert free text query system, direct retrieval from entries in a tree structured hierarchy, and geographic retrieval using a map-walking technique to guide a user to an area of interest prior to data selection.

The second part of Project 901 is concerned with the specification of an EXPERT software package running on a micro (IBM/AT) and used for management of large collections of images stored on optical media.

The two parts are connected by coupling LV-ROM technology to the micro supporting the EXPERT software.

2. THE DOMESDAY PROJECT

1986 is the 900th anniversary of the completion of **The Domesday Book**, ordered by William the Conqueror to establish basic fiscal data, but incorporating a wealth of incidental material on 11th century life. For nine centuries, **The Domesday Book** has been valued as a benchmark in history against which to measure change and development.

To coincide with this important anniversary, BBC Television was asked to consider making a series of documentary programmes on a Domesday theme. It was soon realised that William the Conqueror's problem persists today. Whereas raw data exist about almost any aspect of our lives, it is by no means easy to gain access to them and distil them into useful information. Many statistics are in printed form, housed in widespread archives, libraries, research departments and administrative offices. Even the data available in automated form are kept in large-scale computer systems which are expensive and difficult to explore.

In other words, there is a need today for our own Domesday Book, a convenient means by which inquirers of all types can access the information relevant to their lives.

To meet this challenge the BBC instigated the Domesday project in collaboration with Philips Electronics, Logica and Acorn Computers. The task was to compile a modern Domesday Book using the latest technology. The medium chosen for the 'book' was interactive videodisc.

2.1 Interactive videodisc

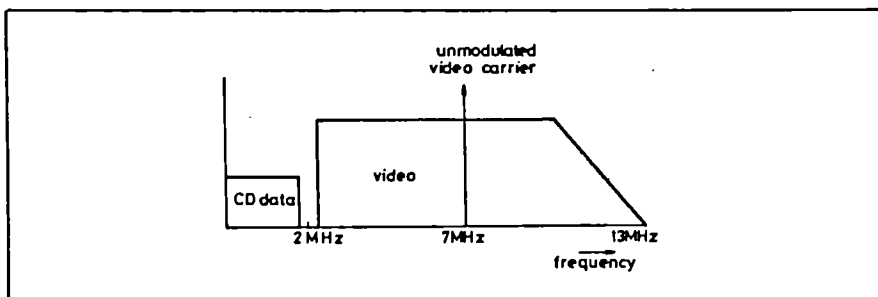
Interactive videodisc is a system in which television pictures and sound are retrieved selectively from a videodisc player by a microcomputer responding to the needs and instructions of the user. The technique has been in use for some time, primarily for educational and point-of-sale purposes.

The Domesday Project takes the idea of the interactive videodisc a step further because it not only requires the storage and retrieval of very large numbers of pictures, but also needs to store vast amounts of data which can be displayed by the computer as text or graphics. There is also a need to store the computer program which will be used to retrieve the pictures and data, and process them according to instructions from the user.

Essential features of the system are rapid access to pictures and data, and the ability to use still frames for long periods. As many of the pictures would be maps, it was important that the player should be capable of very good resolution. These and other factors pointed to the choice of the Philips Laservision System.

A large amount of computer data needed to be stored. Philips' solution to this problem was to use techniques closely allied to compact disc recording and to add a digitally modulated signal below the lower sideband of the frequency-modulated video signal.

Fig. 1 shows the resulting spectrum. The amplitude of the data signal clearly has to be large enough to allow error-free recovery but not so large as to cause visible intermodulation products. This has not in fact proved to be a serious problem. The system is known as LaserVision read-only memory (LV-ROM).



1 Spectrum of recorded video and data

If no audio is required, over 300 Mbytes of data may be stored on a single side of a LaserVision disc. The Domesday Project is producing a double album of such discs, containing data, sound, still and moving pictures.

2.2 Domesday discs

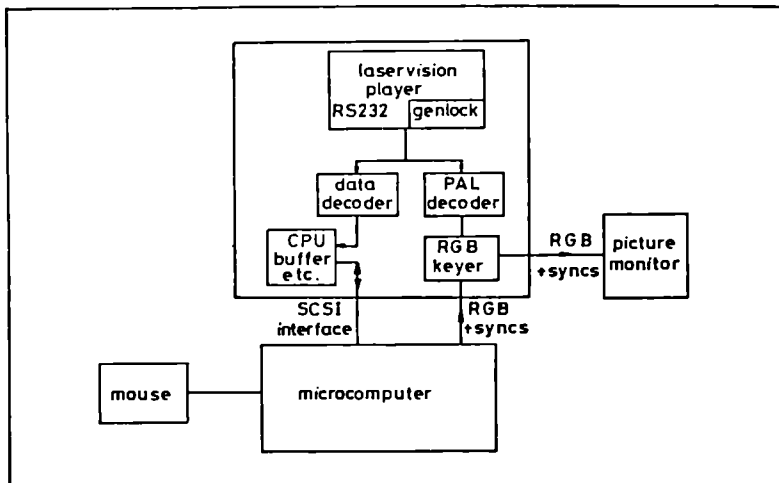
The first Domesday Disc, Community Disc, will primarily contain the bulk of the information collected by schools and community groups and as such has been named 'the people's database'. A major feature of this disc will be access to over 25 000 maps. Most of these are Ordnance Survey maps at a variety of scales. There will be augmented UK and regional maps at the top level and floor plans of special local features (some of them selected and produced by local groups) at the bottom level. The disc provides a facility for 'map walking' and zooming in on areas of interest to the user.

The second Domesday Disc, the National Disc, will include contributions from major national data sources giving a social, cultural, economic and environmental picture of the UK in the late 20th century. there will also be approximately 40 000 photographs. These will be acquired from museums, archives, picture agencies and from the general public via a national photographic competition.

The Domesday Discs will also carry all the indexes and microcomputer software to enable users to retrieve, combine, compare and present the Domesday material in a variety of displays on the monitor screen. In addition, there will be comprehensive help facility and a computer-generated gallery allowing a visual index to the picture sets on the disc.

2.3 Advanced interactive video system

To play the discs there will be a configuration of hardware as shown in Fig.2. This configuration comprises a BBC Master series microcomputer with a special data interface, an RGB monitor and the Philips Laservision player VP 415.



2 Block diagram of system

Communication between the player and a controlling computer is via SCSI interfaces. The player can be used with ordinary Laservision disc containing audiovisual information, or LV-ROM discs, which contain data as well as audiovisual information. These data take the place of the audio channel on some or all sections of the disc.

The player allows all the facilities of the Laservision system to be controlled by a computer. In this way, the computer can control the picture, sound and LV-ROM data. Play is controlled by picture numbers, chapter numbers, autostops or a computer program. Communication with the player is achieved using either a special code known as F-code, or LV-DOS commands.

The F-code instruction set enables commands to be sent (as ASCII characters) to the player, some of these commands causing responses to be returned to the computer. For example, the code F100R means go to frame 100 and display as still frame. LV-DOS commands allow the player to be used as an LV-ROM memory device in a computer system. Both data retrieval from disc and player control are possible. Using F-code commands, the player can participate in an interactive program with any computer system that is loaded with the necessary program.

The player allows the best possible picture quality to be obtained from a Laservision disc by employing a built-in PAL-RGB decoder. Within the Laservision format, video information is stored on the disc in PAL encoded form. This can cause problems when the disc is played in 'still frame' or 'slow motion', or any other non-standard playing mode, because the PAL 8-field sequence is disrupted.

In order to correct this sequence such that a monitor can understand it and reproduce correct colour, many players incorporate a 'PAL modifier'. This piece of circuitry restores the PAL sequence but, in doing so, reduces the video bandwidth, and introduces other unwanted effects, e.g. ringing.

The player tackles this problem by employing a fast-locking PAL-RGB decoder. Having this device built in allows its characteristics to be fully optimised to give the highest possible picture quality from the disc, even in non-standard playing modes. The result is an RGB output giving the full 5 MHz video bandwidth in all playing modes. The benefit is particularly valuable when viewing video material such as maps with fine text.

The player contains an internal sync pulse generator (SPG) which may either free-run, or in the presence of a suitable reference signal, lock itself to the external reference (genlock).

The SPG provides freshly generated line and field sync pulses at the player's video output at all times. Following the decoding process from PAL to RGB, fresh sync pulses are inserted into the RGB signal, which is available at the Euroconnector socket. Therefore a stable output from the player is guaranteed at all times.

A charge-coupled device (CCD) time-base corrector is employed to provide correction of timing errors always present in the signal replayed from the videodisc. This replaces the more traditional tangential mirror (mechanical method) allowing for a smaller, lighter optical system. This reduction in mass allows the scanning lens assembly to track the disc faster and thus reduces picture access time.

The player also incorporates an 'instant jump' feature. Essentially this means that the radial mirror which points the laser beam at the required disc track can be made to 'twitch' and therefore jump a predetermined number of tracks (max. 50) in either direction during the vertical interval (field flyback time). This gives the effect of an instant search to the required picture - almost as if it were immediately adjacent to the current picture.

This feature is valuable in, for example, 'map-walking', where each picture contains a map of an area and each successive picture shows an adjacent area. The user can scan across map boundaries with no black picture between maps while the player searches for the next picture.

An important feature of the system is the ability to mix graphics generated by the microcomputer with the picture from the videodisc. Among other things this allows a mouse-controlled pointer and menus to be superimposed on the picture, grids to be added to maps and graphics to be mixed with pictures.

This can only be done if the graphics output of the computer conforms with 625-line television standards and if the disc player and the computer can be synchronised. The BBC Master Series computer made by Acorn is almost alone in generating a 625-line waveform although its field synchronising waveform, in common with most microcomputers, is a very much simplified version of CCIR recommendations. To achieve synchronisation, the videodisc player is genlocked to the microcomputer.

The computer has both RGB and PAL encoded video outputs. Experiments showed that much better results in terms of resolution and hence legibility can be achieved if the RGB output is used rather than the coded output, and the mixer uses the RGB output of the computer as one input and the decoded signal from the videodisc as the other. The largest of the R, G or B signals from the computer can also be used as a key signal so that five modes of mixing are available:

- ◆ player only
- ◆ microcomputer only
- ◆ hard key of microcomputer output over videodisc picture
- ◆ transparent mixing of microcomputer graphics over videodisc picture
- ◆ Highlight: videodisc pictures at full amplitude in area defined by microcomputer graphics and reduced in amplitude elsewhere.

The majority of the video material on the discs is single frame. Some 100 000 frames have been recorded during the preparation of the discs. The sources to be recorded included Video Rostrum Camera, Slide Scanner and Quantel paintbox. Considerable care had to be taken in a number of areas. First, videodisc production is extremely sensitive to drifts and variations in sync - subcarrier phase. Secondly, viewing a large number of single frames requires that they be accurately colour balanced. Both these requirements meant careful line up of equipments, particularly as the material was assembled over a period of 15 months. This was also necessary as the output from the microprocessor is overlaid onto the output of the player. For much of its operation the system uses a micro-generated icon to point to material in the videodisc picture. Therefore a large number of video frames had to be recorded to an accuracy of better than 100 ns in the horizontal plane.

Finally, the material has to be recorded to a positional accuracy of one frame on the tapes and the time code of the recording conveyed to the Domesday software. Special control systems were devised for this purpose. The task of the video recording was carried out by the BBC's Open University Production centre, which has considerable experience in this technique.

2.4 Software

The Domesday Project required special software for data collection, for recording it on the disc, and for its retrieval and display using the micro-processor.

To facilitate the task of collecting large amounts of textual data from schools and communities from all over the country, a set of data collection software was sent, on a floppy disc or cassette tape, to each group contributing items to the Community Disc. The software, written in Basic, allowed people to type their text into either a BBC or RML microcomputer, for subsequent output on a floppy disc or cassette. The discs and tapes containing the text were then returned to the BBC.

At BBC offices in Ealing, West London, the task of assembling the Domesday data in preparation for mastering onto the videodiscs was performed. All the data were held on a DEC VAX 11/750 minicomputer. Community Disc data was initially transferred from the discs and tapes to a Multics computer at Loughborough, and then to the VAX at Ealing. National Disc data emanated from several sources, including universities and government departments all over the country, and were sent to Ealing on magnetic tape for transfer to the VAX.

BBC programmers have written premastering software in Fortran 77 which processes the data into standard format for recording on the videodiscs. The software has to process large amounts of data, which arrive from the contributing groups in a variety of formats, and also to construct index files to the data. These index files are also recorded on the videodisc and allow subsequent retrieval of data when the Domesday system is being used.

The software which enables a user of the Domesday System to examine the data on the videodiscs - the retrieval software - was written by Logica in the BCPL language and it executes in the BBC Master Series microcomputer which controls the system. In general, today's computer systems are designed around the concept of storing large volumes of data on magnetic discs and a special piece of system software was written by Acorn to allow the retrieval software to read data from a videodisc. It is called the video filing system and resides in a ROM chip in the microcomputer. The retrieval software performs all its videodisc accesses through the video filing system.

The retrieval software itself resides on the videodiscs. When the system is started up, the controlling section of the software - the 'kernel' - is read into the microcomputer's memory from the videodisc. The kernel stays in memory permanently while the system is in use and decides which other parts of the software to load from the videodisc.

The software can only execute while it is in the microcomputer memory and there is insufficient memory to store all the retrieval software. Therefore overlaying is used: only the required software overlay is read from the videodisc ; when a new overlay is required the current overlay is discarded and the new overlay is read into the same memory area.

The retrieval software is designed as a finite state machine system. At any instant the software is in a definite state, depending on what action the user is performing. A 'state table' held permanently in memory tells the software how the user may enter any state from any other state. For example, if the user is looking at maps on the Community Disc, the software is in the map state. If the user then requests help by pressing the help key, the software reads the state table to determine how the requested transition is performed and then enters the help state. In this way, the software can control the user's progression through a complex series of actions.

2.5 The Domesday system

The result of the technology is the Domesday System, but what does it look like ? To describe it in detail requires an article to itself.

Essentially, a user sits at a computer terminal, but the visual display is a composite of videodisc picture overlaid with the microcomputer outputs. By calling up menus, maps and other visual indexes, access may be made to the vast amount of pictures, text and data stored on the discs. No two people can be expected to operate the system in the same way, each using it to provided their particular requirements. It is thus fully interactive and a major step forward in this developing area of audio/video technology.

2.6 Advanced retrieval features

By the end of the first year of ESPRIT project 901, in December 1986, the Domesday system had been developed and tested to product status. It was launched by BBC Enterprises Ltd in November 1986. Sales since the launch of the system have been encouraging and have been split approximately equally between the education and commercial sectors. In the second year of the project, more advanced retrieval features are being added to the system. these are described in the following paragraphs.

Acorn computers Ltd have developed a new and extremely fast processor device called RISC (Reduced Instruction Set Computer) and are using the device in their new range of Archimedes microcomputers. ESPRIT 901 is using the power of the new machine to develop the Domesday retrieval facilities into an intelligent 'expert' system which can provide constructive help to a user who is attempting to perform research using the Domesday data. The RISC processor provides more power than many minicomputers and has access to 4 megabytes of memory. It is being used by Logica to implement a 'Domesday Intelligent Statistics Helper' (DISH). DISH will lead a user through the statistical procedures he wishes to use and will indicate the sequence of steps he should perform in order to achieve the desired result. It will build up internal pictures of users which are used to restrict the amount of help given to users who do not require it ; this process is termed 'user modelling'.

The Domesday system allows a user to 'tour' an area by showing still video of United Kingdom Ordnance Survey maps at various scales. A zooming feature is provided which shows maps of one area at 5 different scales. This feature is being developed in year 2 of the project into one which can manipulate a digitally stored map image and provide a continuous zoom and pan facility. The Commodore Amiga microcomputer is being used for this process because tools are available for the Amiga which enable a video image to be digitised and stored in the main memory of the computer.

Authoring systems allow a user to create a 'customised' sequence of progression through a videodisc. One of the advanced retrieval facilities being added to Domesday is the ability to create a presentation of graphical and video displays made during a Domesday session. The presentation can be stored on a floppy disc and synchronised with an audio commentary stored on a cassette tape. The resulting presentation can be replayed at various speeds and repeated many times. This system is invaluable for education where effective presentations prepared in advance are vital for conveying the teacher's message.

With the addition of these extra facilities, the Domesday system is transformed into an intelligent and easily manipulated visual database system.

3. EXPERT SYSTEM TO PROVIDE IMAGE MANAGEMENT FACILITIES

The ESPRIT Project 901 is actually defined by two objectives:

- Implementation of the LV-ROM technology via the DOMESDAY product
- Specification of an EXPERT system running on a micro and used for management of large collections of images stored on optical media.

We shall now present the second point, which is being considered by SEP with BMvD and CRIN, as subcontractors.

Particular effort is being made to connect the two objectives by coupling LV-ROM technology to the micro supporting the EXPERT software.

3.1 WHY an EXPERT system on a micro ?

- . high potential of the image market
- . large implantation of the micro in public and industrial activities
- . non specialist end users, within electronic image data base management and processing systems
- . improvement of high reliability optical storage technologies for general public use.

3.2 HOW to approach an 'Intelligent Retrieval System' ?

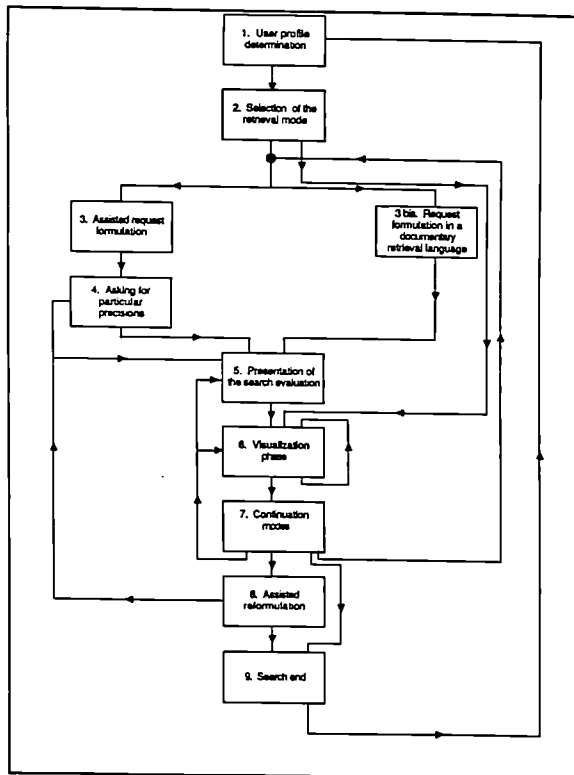
- . 'STATE OF THE ART'
 - image storage technologies
 - advanced expert systems
 - Data base management systems.
- . 'SPECIFICATION'
 - image data structures
 - dialogue and piloting modules
 - hardware architecture
 - display handling functions.
- . 'MARKETING'
 - technical feasibility
 - economical feasibility.

'State of the art' has been already worked out and has shown the need for:

- . videodisc technology for large interactive image database
- . a dedicated advanced expert system applied to images.

3.3 Specification of such a dedicated expert system

- . Herunder are schematized the results of our study, concerning the logical concept of the EXPERT system.



Hardware architecture

Because of the large distribution of IBM micros in the market, we have chosen, after technical studies, to work with such equipment (in particular with an IBM/AT).

The use of the actual 'IMAGEUR DOCUMENTAIRE' SEP product is well accepted because of its adequacy for such image applications. (the 'Loader' (see below) is a particular software package written at SEP for activating diverse specific tasks).

With the both graphs (see below), we are able to demonstrate, today, that a standard IBM/PC hardware architecture is correct and efficient for such use.

SPEC HARDWARE ARCHITECTURE

IBM - PC/AT COMPATIBLE

UP TO NOW

SYSTEM -----> 70 Ko		
LOADER -----> 30 Ko		
'IMAGEUR' 80 Ko	SGBD 150 Ko	DATA N. IMAGE
512 Ko		

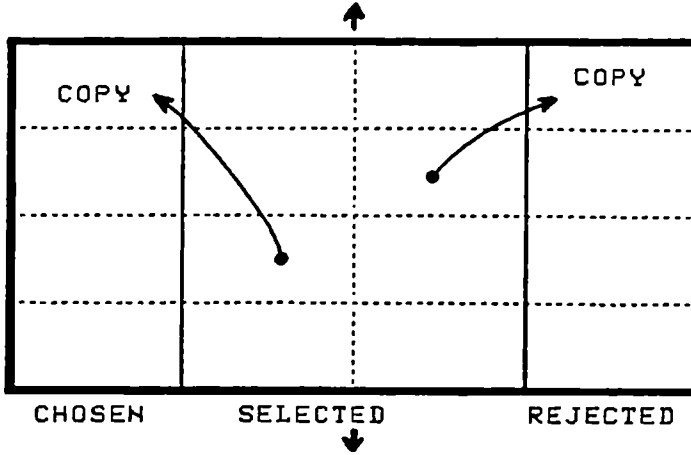
EXPERT SYSTEM

SYSTEM -----> 70 Ko		
LOADER -----> 30 Ko		
'IMAGEUR' 80 Ko	EXPERT 250 Ko WITH RULES (~50)	SGBD + thesauri 150 Ko
VIRTUAL DISK LISTS OF IMAGES (VIRTUAL DISK > 1 M#)		

. Handling functions

A very small sub-set of 'Imageur Documentaire' handling functions is used for EXPERT application (for the non-specialist end user).

Below are represented the general functions and configuration used within the 'MOSAIC IMAGEUR' concept (16 independant displayed images on the 'MOSAIC' screen).



FUNCTIONS : - Display (selected images after documentary request)

- Copy (selected image to :
 - chosen zone
 - rejected zone)
- Backward
- Forward

3.4 Marketing

We are aware that a mock-up of a dedicated expert system based on a micro is not a finalized product. So we are working today on a marketing analysis, in order to draw out the main technical and economical rules to optimize impacts on actual and potential markets in that field.

ANNEX 1

DESCRIPTION

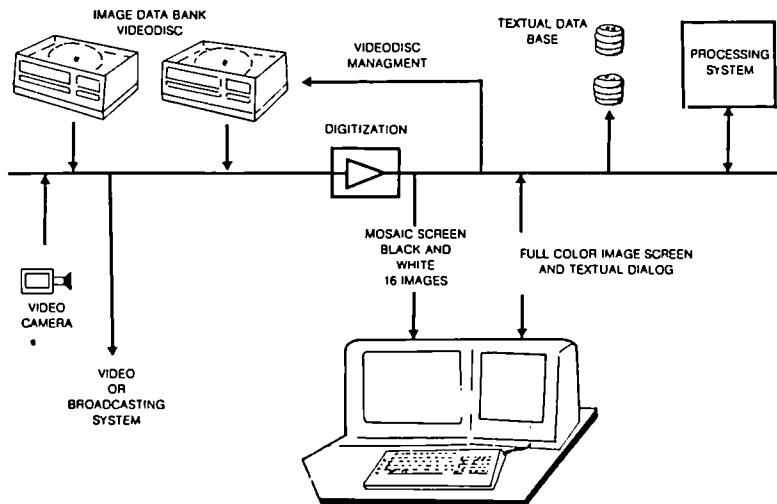
The "IMAGEUR DOCUMENTAIRE" is based on an IBM PC/AT or compatible.

It can pilot a battery of videodiscs of 54000 images each.

All the visualizations are made on 2 screens:

- a high resolution screen for the mosaic with 2 kinds of visualization:
 - . a four images mosaic (black and white)
 - . a sixteen images mosaic (black and white)
- a screen for alphanumeric dialog and full color image.

It includes a Data Base Management Software, Micro-Questel, TELESYSTEMES, and a specific hardware and software for all the handlings of images developed by SEP.



OPTIONS

- It can be connected with an external Data Base Management System.
- It can pilot a battery of Digital Optical Discs.
- Different kinds of outputs can be provide according to the application.

FIELDS OF APPLICATIONS

At the time of the explosion of the 'Image Communication', the 'IMAGEUR DOCUMENTAIRE' will be of a great help for those who want a high level of profitability for their images collection:

- First of all, by the organization of image banks on stand-alone work stations or connected with a network.
- Then, by the putting into operation of a strategy of distribution adapted with a specific application.

This is SEP's knowledge to provide you a support for these two steps, owing to a team of images specialists on various fields as well in press, cinema, television, museums as in satellites, cartography, medicine or any kind of scientific images.

Project No. 563

PHOTOVIDEOTEX IMAGE COMPRESSION ALGORITHMS - TOWARDS INTERNATIONAL STANDARDISATION

Graham P. Hudson

British Telecommunications plc
British Telecom Research Laboratories
Martlesham Heath, Ipswich IP5 7RE
United Kingdom

From the ten photographic compression techniques that were originally being studied by the PICA consortium two have been developed as candidates for international standardisation. These provide progressive build-up with very good quality results being produced at a compression of over 20:1 (an average of 0.75bit/pixel). Real time decoders are being implemented for a demonstration at the final standards selection meeting in January '88. There is a good possibility that the PICA work will be incorporated in a standard.

1. INTRODUCTION

1.1. Project Aims

Photographic pictures have a high spatial pixel resolution and each pixel is represented by three multilevel colour components. A studio quality video picture has 720X575 pixels where each pixel is represented by one byte of luminance (black and white) and two bytes of chrominance (colour) information. The colour resolution is horizontally half that of the luminance. The digitised picture contains a total 818kbytes of information.

The aim of the ESPRIT 563 project is to develop a technique for compressing photographic data to enable photovideotex services to be possible on the 64kbit/s Integrated Services Digital Networks. The goal was to reduce the data by a factor of 16:1 (an average of 1bit/pixel) whilst maintaining acceptable quality. A full frame photograph could then be stored in 52kbytes of memory and be transmitted in 7s. The complexity of the technique, however, must be such that it can be economically decoded in real time by a terminal. The technique was then to be submitted to the international standards bodies.

The consortium consisting of seven partners *(see footnote) with telecommunication, broadcasting and industrial experience is now in its third and final year. As reported at the ESPRIT '86 Technical Conference, during the first year of the project's test criteria were set and 10 potentially useful techniques were evaluated and developed. The principles of the techniques and their characteristics are described and results meeting the project's requirements are shown in [1].

1.2. International Standards Scene

The focal point for photographic standardisation is now the ISO/CCITT Joint Photographic Experts Group (JPEG). The work plan of the project both in terms of technical requirements and time scale has been

BTRL - British Telecom Research Laboratories, Ipswich, United Kingdom (Prime Contractor)
IBA - Independent Broadcasting Authority, Winchester, United Kingdom
KTAS - Copenhagen Telephone Company, Copenhagen, Denmark
DNL - Dr Neher Laboratories, Leidschendam, The Netherlands
CSELT - Centro Studi E Laboratori Telecomunicazioni SpA, Turin, Italy
CCETT - Centre Commun D'Etudes De Telediffusion Et Telecomunications, Rennes, France
Nixdorf - Nixdorf Computer AG, Berlin, West Germany

greatly influenced in the last year by the activities of the JPEG. During the year JPEG has set out a specification for a compression technique and a three-stage selection process resulting in a technique for standardisation in January '88.

Although photovideotex is the primary motivation of the group the techniques being developed are also applicable to still picture television (teleconferencing and surveillance), phototelegraphy (newspaper picture transmission) and photographic content coding of office documents. The needs of these other services have been made known to the JPEG by the CCITT and other liaison members (see figure 3). These are reflected in the requirements laid down which are broader than would be needed for videotex.

1.3 Recent Project Work

Rather than present all techniques to the standards bodies, at the end of the second year ('86), to ease the standardisation process and to limit the competition between partners in the international forum, it was decided to try to reduce the number of techniques under development. It was apparent that some techniques did not and would not meet the compression/quality requirements in the timescale for international standardisation and so could be eliminated. The techniques under development that could meet these requirements could be classified into two categories, and to bring about the reduction and to develop those remaining techniques two sub-groups were set up: one to produce a predictive technique and the other to produce a transform technique. The membership of these is shown in figure 2.

Name Of Technique	Developer	Predictive Sub-Group
Adaptive Block Truncation Coding (ABTC)	BT	BT - G Hudson, M Beaumont, C Nightingale, D Tricker*, N Street
Recursive Binary Nesting (RBN)	BT	IBA - R Vivian
Adaptive Differential Pulse Code Modulation (ADPCM)	IBA	DNL - C Baaijen, J Brahms, H Reitsema H Van Helden
High Correlation Transform (HCT)	KTAS	Transform Sub-Group
Low Correlation Transform (LCT)	KTAS	
Vector Quantisation (VQ)	DNL	CCETT- A Leger*, C Guillemot, F Lemoine T N'Guyen
Vector Quantisation Using Peano Scan (VQps)	CSELT	KTAS - H Poulsen, T Dam, B Niss, J Vaaben CSELT- M Guglielmo, L Chiariglione, G Conte, L Conte
Colour Vector Quantisation (CoIVQ)	CCETT	principal representatives of each partner is given first
Adaptive Discrete Cosine Transform (ADCT)	CCETT	
Hierarchical Predictive Coding (HPC)	Nixdorf	leader of sub-group is denoted by *

Figure 1, Techniques Originally Developed By The ESPRIT Consortium

Figure 2, Membership Of The Two Development Sub-groups

This paper gives the requirements of a photographic compression technique laid down by the ISO/CCITT JPEG. The transform and predictive techniques developed by the consortium and presented to the JPEG are described and the results of the JPEG initial selection process which took place in June '87 are given.

2. INTERNATIONAL STANDARDS REQUIREMENTS

2.1. Committee Structure

WG8 was set up in ISO TC97/SC2 (Data Processing/Character Sets and Information Coding) in 1983 to deal with all types of picture coding. WG8 took as its starting point for photographic coding the preliminary functionality and coding given in the CEPT Videotex Recommendation [2] but technological advance has made better performance possible.

The ESPRIT PICA group has actively participated in WG8 and presented the results of its first year's work

on ten different compression techniques in March '86 [3]. The main activity in WG8 then concentrated on the selection of a new high performance compression technique. Japan responded to this European initiative by setting up a national coordination committee known as the Natural Image Standards (NIS) group consisting of members from PTTs, universities and industry (most of the well known names in consumer electronics). Later that year they produced a paper [4] on four techniques for consideration as an international standard.

Also in March '86 the CCITT SGVIII group that deals with Telematic Services set up a special rapporteur's group to consider New Image Communication (NIC). They were particularly concerned with finding common techniques for different applications within future integrated networks [5]. NIC and WG8 soon joined forces to form the Joint Photographic Experts Group in the autumn of '86. JPEG then had the responsibility for selecting a photographic compression technique for ISO and CCITT standardisation. The first task in the selection process was the establishment of a specification for the technique.

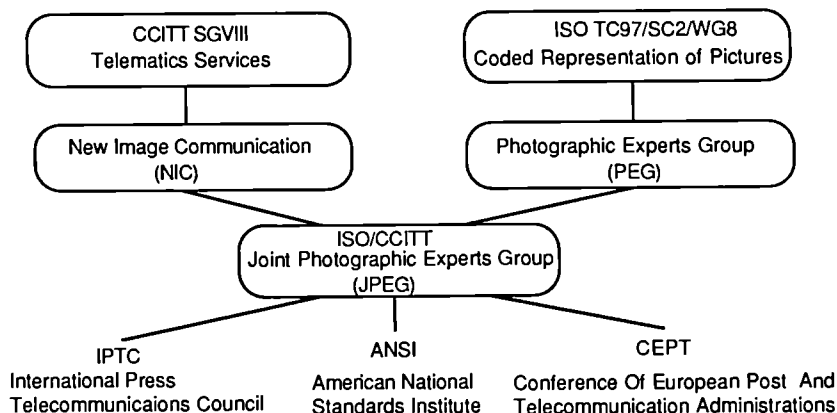


Figure 3, Relationship Between The International Standards Bodies

2.2. International Requirements

The main objective of the JPEG is to find an efficient and economic coding syntax for multi-level monochrome and colour pictures. Quantitatively JPEG chose the same requirements as the PICA project, that the technique should compress a studio quality picture of 720 X 575 X 16 bits by a factor of 16:1 and should be capable of being decoded in real time at 64kbit/s. Although all performance figures are quoted at the studio resolution the technique should also be applicable at resolutions for 256 X 256 to 2048 X 2048.

Another feature considered mandatory was progressive picture build-up. This provides for a crude picture quickly which is subsequently improved in several stages to a final very good quality. This feature allows the viewer to get an idea of the overall picture content and to decide whether or not to wait for the remainder of the data to produce a better quality picture. It is a particularly desirable feature on lower data rate networks such as the PSTN. A compression of 64:1 (0.25 bit/pixel) was agreed for the initial stage producing a picture in under 2s on the ISDN. A final stage at 4:1 (4bit/pixel) compression was then defined for a picture indistinguishable from the original.

A progressive build-up scheme with more stages can be beneficial as it can provide for a smoother picture build-up and therefore better subjective appearance. There are some applications such as medical images, newspaper pictures and surveillance pictures where it can be necessary to produce a final image identical to the original. Such a technique is referred to as lossless, reversible or exact.

Although progressive build-up is desirable for soft displays, for transmission of pictures between computers or from computer to hardcopy device a single-stage sequential build-up is desirable.

2.3. Standardisation Schedule

The CCITT recommendations can only be ratified by a plenary that meets at the end of every 4 years. The present period ends in 1988 and so this became the deadline for a compression technique to be selected. At the first JPEG meeting in November '86, a schedule of three selection meetings was agreed: the first for techniques to be registered as candidates, the second to make an initial reduction of the number of techniques and the third to select the technique for standardisation. These meetings were/are to be held in March '87 (Darmstadt), June '87 (Copenhagen) and January '88 (Copenhagen). Copenhagen was chosen for the final two selection meeting because of need for subjective testing facilities which KTAS had installed for the PICA project.

3. PROGRESSIVE RECURSIVE BINARY NESTING

3.1. Principles Of PRBN

3.1.1. Introduction

PRBN is a progressive build-up version of the original RBN invented by BT and described in [1], but with several new significant refinements added. The principle of the technique is to vary the spatial sampling density adaptively according to the picture content. More samples are coded in 'busy' (high detail) areas than in 'quiet' areas. For coding efficiency the differences between sample values are coded and quad tree coding is used to address sample points. From these sample points the prediction is made that the area within the block is a bilinear interpolation of the corner points. To achieve indistinguishable and identical pictures differential coding (DPCM) from the previous stage is used. Two patent applications have been filed by on ideas embodied in the PRBN technique [6] [7].

3.1.2. Sub-sampling

The encoding process commences by coding the four corner points (ABCD in figure 4) of a large block of the picture. This could be the whole picture but in practice is likely to be one of a number of large rectangular blocks into which the picture has been divided. An acceptance test is then performed on this block to ascertain whether the prediction is a sufficiently accurate representation of the original. If acceptable then a 0 is coded indicating no further information is to be coded for the block and the process is complete. If not then a 1 is coded and the block is sub-divided into four and five new corner points (PQRST in figure 4) are coded.

If a sample point has already been coded in an adjacent block it is not repeated. New sample points are coded by entropy coding the quantised differences between actual and predicted (by interpolation) values. The process is continued until every pixel is coded or until the maximum sub-division (minimum block size) is reached. The latter increases compression at the expense of resolution. To avoid a spurious pixel sample being coded a filter is used to choose the best of the bunch of pixels in the corner.

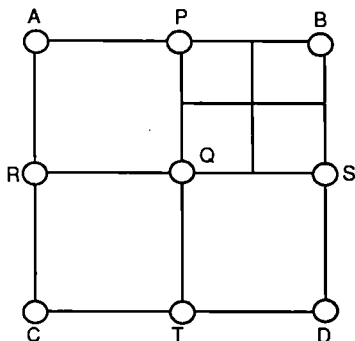


Figure 4, PRBN Sub-sampling Structure

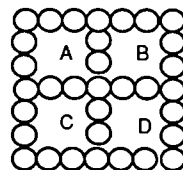


Figure 5, PRBN Recursive Interpolation

3.1.3. Interpolation

At the decoder the picture is reconstructed using two dimensional interpolation between the sample values. Several different methods of interpolation have been studied including a diagonal interpolation. A recursive bi-linear technique was chosen because of its simplicity and freedom from artefacts.

Firstly the pixels along the edges of the area are filled by linear interpolation from the corners as shown in figure 5. Along an edge the mid-point pixel is taken as an average of the two corners and then this process can be repeated recursively until all the pixels along the line have been filled. Having completed the edges, the block is divided into 4 and the new internal edges are filled. Again this process is repeated recursively until all the pixels on the display are filled.

Adjacent blocks may have already been coded and the edges may contain actual rather than predicted values. In this case the existing values are used. As adjacent blocks have common boundary pixels there are no discontinuities at block edges. There may be discontinuities in gradient but these are rarely obtrusive.

3.1.4. Block Acceptance Criteria

The block acceptance criteria consist of two stages designed to identify different significant features. The first criterion is the grey level correlation test. All pixels above a certain threshold are marked as error pixels. If a cluster of error pixels is detected then the prediction is considered unacceptable and the block is sub-divided. The threshold can be varied according to average block grey level and size.

It was observed that the continuity of edges and lines were subjectively important and so a second criterion incorporating an edge detector was introduced. Two types of detector have been used - Sobel/Hough and Laplacian. The latter was used to generate results for the boats test picture which includes fine rigging. This criterion is only applied if the first criterion is found to be satisfactory. When a line is detected the block is sub-divided and the acceptance tests are reapplied.

3.1.5. Differential Pulse Code Modulation

To create an indistinguishable picture DPCM is employed using the main interpolated stage and previous pixels of the present stage as the predictor. A linear quantiser is used with intervals equal to half the threshold used in the block acceptance criteria. Another DPCM stage using the indistinguishable stage as the predictor provides a picture identical to the original. A paper describing some of the DPCM work undertaken in the consortium has recently been published [8].

3.2. Features Of PRBN

3.2.1. Decode/Encode Complexity

A major feature of the RBN technique is its potential for decoding at 64 kbit/s using a general purpose microprocessor. Decoding a block consists of two stages - the interpretation of quad tree flag bits and the filling of blocks by interpolation. By far the most time consuming item is the interpolation. The interpolation method described earlier has been optimised for implementation in software. Calculations that have been made for a 1bit/pixel picture with a representative mix of different block sizes for a decoder written in Intel 80286 assembly language suggest that real time operation would be possible with an 8MHz cpu clock.

The encoder is more complex than the decoder because of the acceptance criteria associated with the adaptive subsampling. It is anticipated that with optimised software a full-frame picture could be encoded in a few minutes. The encode time can be reduced by removing the edge detection criterion. For real time encoding additional computing power such as a coprocessors/digital signal processor would be needed.

3.2.2. Picture Build-up

To produce the initial stage of the progressive build-up the amount of sub-sampling is limited. For studio resolution pictures the maximum number of sub-divisions for the first stage is 6 to give a compression greater than 0.25 bit/pixel. (For a 512 X 512 pixel picture a division by 2^6 results in 8 X 8 pixel blocks and for 720 X 575 the minimum blocks are 11/12 X 9 pixels. Note blocks have the same aspect ratio as the whole picture and are not necessarily square.)

For subsequent stages the number of permitted sub-divisions is increased thereby improving the quality and the number of bits per pixel. During the encoding process if the block acceptance criterion indicates a further sub-division is needed the block is flagged for further sub-division on the next stage of progressive build-up. The number of stages is variable depending on the number of sub-divisions allowed for each stage and can be greater than three stages. Results for the initial (0.25bit/pixel) and main stage (0.75bit/pixel) using PRBN on the very difficult Barbara with toys test picture are shown in figure 6.

The RBN block acceptance criteria thresholds have been tuned to provide a very good quality picture at the main stage of 0.75bit/pixel. Further stages are provided by sending DPCM information using the existing picture as the predictor. An indistinguishable picture is created after one stage of DPCM and this requires a further 3bit/pixel. Another stage gives a picture identical to the original for a total of about 8bit/pixel.

The PRBN algorithm can be made to produce a picture in one sequential scan by specifying the quality/compression required and then all the sample points required are sent in one scan. The subjective effect is improved by applying PRBN to a limited area at a time, eg 16 lines vertically.



0.25bit/pixel



0.25bit/pixel



0.75bit/pixel



0.75bit/pixel

Figure 6, Results Using PRBN

Figure 7, Results Using ADCT

3.2.3 Special Features

A special feature of the PRBN technique is resolution independence. The encoding process first of all takes the corners of the whole picture or sub-picture and then proceeds to sub-divide it into blocks. The process is not dependent on the size of the original and does not divide the picture into blocks of a fixed number of pixels. The coded information is not characterised by the input frame store and so the picture is

coded in a resolution-independent way. At the decoder the picture is reconstructed by interpolating according to the number of pixels on the receiver display. It is therefore possible to encode a picture from a frame store of one resolution and display it on framestore of different resolutions. When the frame stores have the same resolution the blocks on the receiver display contain the same number of pixels as the source framestore.

The technique does not have to operate on pictures that are multiples of particular block size eg multiples of 8X8. The scheme's progressive sub-sampling means that information suitable for a low resolution display is transmitted first, followed by further data for a higher resolution terminal.

4. ADAPTIVE DISCRETE COSINE TRANSFORM

4.1. Principles Of ADCT

4.1.1. Introduction

The ADCT technique described in [1] has been radically modified taking into account ideas developed on the HCT/LCT technique. The result is a progressive build-up ADCT producing excellent pictures below 1bit/pixel (see figure 7). The DCT is applied to 8 X 8 pixel blocks of the picture and results in a matrix of 64 spectral domain coefficients. The coefficient in the top left hand corner of the matrix is the DC coefficient and represents the mean pixel value of the block. The other terms are the AC coefficients which are of increasing frequency moving away from the top left. The value of the coefficient represents the amplitude of the cosine function for that particular frequency. Compression is achieved by varying the accuracy of coding the coefficients and by not coding some coefficients. At the decoder an inverse DCT is applied to reconstruct the pixels of the block.

4.1.2. Coefficient Selection And Quantisation

The choice and accuracy of coding the coefficients in the PICA ADCT is based on the results of psycho-visual experiments. The coefficient values are divided by the threshold values shown in figure 8 and the result is then quantised. The low frequency components have been found to be most important and are coded with greatest accuracy but some of the high frequencies can be ignored and are set to zero.

Normally the higher frequency chrominance coefficients are coded using a very coarse quantisation. However the consortium found that better representation of the chrominance coefficients adds greatly to the subjective quality of the reconstructed image with only a small penalty in bit rate. When the coefficient are quantised even some originally non-zero coefficient vanish.

16, 11, 10, 16, 24, 40, 51, 61,	1, 2, 6, 7, 15, 16, 28, 29,
12, 12, 14, 19, 26, 58, 60, 55,	3, 5, 8, 14, 17, 27, 30, 43,
14, 13, 16, 24, 40, 57, 69, 56,	4, 9, 13, 18, 26, 31, 42, 44,
14, 17, 22, 29, 51, 87, 80, 62,	10, 12, 19, 25, 32, 41, 45, 54,
18, 22, 37, 56, 68,109,103, 77,	11, 20, 24, 33, 40, 46, 53, 55,
24, 35, 55, 64, 81,104,113, 92,	21, 23, 34, 39, 47, 52, 56, 61,
49, 64, 78, 87, 103,121,120,101,	22, 35, 38, 48, 51, 57, 60, 62,
72, 92, 95, 98, 112,100,103, 99,	36, 37, 49, 50, 58, 59, 63, 64,

Figure 8, ADCT Psycho-visual Thresholds For Luminance Figure 9, ADCT Zig-Zag Scan Path Matrix

4.1.3. Coding Of The Coefficients

A different strategy is used for coding the DC coefficients which are always present, from that for the AC values which may be zero.

The distribution of DC values is normally very broad and even with entropy coding many bits are required. The DC values of a given block, however, often only differ slightly from that of the preceding block. Hence the differential coding of the DC value exhibits a much narrower distribution and entropy coding requires only a few bits.

The AC components of a block are coded in the order given by a zig-zag scan path as shown in figure 9. The low frequency components get low indices and since the coefficient values normally decrease with increasing frequency the procedure tends to cluster vanishing coefficients towards higher indices. The zig-zag scanned matrix exhibits runs of vanishing coefficients especially of higher order. The number of zero coefficients before the next non-zero coefficient is represented compactly by variable length code (VLC) so that the most frequent runs can be coded in very few bits. The coefficient value is then coded.

As the coefficient values are often small requiring only a few bits the number of bits in the next coefficient can be combined into the code for the number of zeroes. For example, if there are three zeroes and the first non-zero coefficient is represented by 2bits then the run length is represented by the vector (3,2) and the actual code transmitted to line is 2bits. Huffman coding compresses data well because the most common runs (eg 0,1 1,1 2,1) will occupy very few Huffman code bits and very few coefficient code bits. Further, since 0 need not be represented and the number of bits in the code is known, a new representation for the n possible values of the coefficient can be produced as shown in figure 12. Each coefficient is then represented by one bit less than would normally be used for a coefficient of that magnitude.

4.2. Features Of ADCT

4.2.1. Encode /Decode Complexity

The computation of the transform in the encoder and the inverse in the decoder involves real arithmetic and neither cannot be performed with present day microprocessors at 64kbit/s. Earlier in the project a real time DCT decoder was implemented using a Digital Signal Processor chip (NEC 7720) for another DCT scheme (Chen and Smith). It is believed that the new ADCT scheme could be decoded using modern DSP chips such as the TMS 320 series. There are now becoming available dedicated DCT transform chips which may be used in a terminal.

The consortium has also been involved in the development of a video rate DCT decoder. The technique which uses a patented distributed arithmetic process [9] and a VLSI device is currently being designed. The picture would then be stored in the terminal in compressed form which would dramatically reduce the amount of memory required in a display terminal. The decoding of the variable length data to generate the transform coefficients would be performed by a microprocessor.

The encoding and decoding processes are not symmetrical since the encoder also has to generate the VLC tables. However, it is believed that with the impetus of face to face ISDN services hardware will be developed to make real time encoding possible.

4.2.3. Picture Build-up

The coding scheme is optimised to give a set of quantised coefficients that will produce a subjectively excellent picture below 1bit/pixel. A sequential build-up is readily produced by coding all the coefficients of one block followed by the next block.

To produce a progressive build-up the DC coefficients for all blocks are coded followed by the AC coefficients in ascending frequency. For the initial 0.25 bit/pixel stage the DC coefficients are transmitted for luminance and chrominance and the first two non-zero AC luminance coefficients. The main stage is then formed by transmitting all the remaining non-zero luminance and chrominance coefficients. The third 'indistinguishable' stage is produced by DCT coding the difference between the main stage picture and the original and this gives the required overall bit rate of 4bit/pixel. The DCT technique does not naturally provide for reversibility, but if essential an identical picture could be created by sending the difference between the DCT picture and the original using DPCM.

4.2.4. Special Features

Presently there is a lot of international effort to develop low bit rate 64kbit moving picture codecs ('talking head' - small screen videophones) and one leading contender is a DCT scheme. It is therefore possible that a terminal equipped with DCT hardware would have some inherent compatibility between moving and still picture services.

5. INTERNATIONAL STANDARDS SELECTION PROCESS

5.1. Selection Procedure

The twelve techniques listed in figure 10 were registered as candidates for international standardisation at the Darmstadt meeting. Each candidate had to provide a complete description of the principles of the technique and an explanation of how the technique met the technical requirements. Results had to be produced for four test pictures - Zelda, toys against a blackboard, boats and lighthouse and Barbara with toys. High quality digitisations were produced by the IBA in COST 211 magnetic tape format and distributed to proposers to enable a common source to be used. For the initial selection meeting a digitised picture had to be produced for the three stages of the progressive build-up on a magnetic tape. Also candidates had to provide the executable code that generated the pictures in either VAX or IBM PC format to enable their results to be verified.

Name Of Technique	Proposer	Parameters Of The Technique	Weighting
Discrete Cosine Transform With Vector Quantisation (DCTVQ)	NIS	Subjective Quality Of Each Stage	10%
Adaptive Discrete Cosine Transform (ADCT)	ESPRIT	Progressive Build-up	25%
Adaptive DCT And Differential Entropy Coding (ATADEC)	ANT	3rd Stage	5%
DCT With Low Block To Block Distortion	DEC		
Block List Transform (BLT)	AT&T	Decode Complexity	15%
Progressive Coding Scheme (PCS)	NIS	Encode Complexity	10%
Progressive Recursive Binary Nesting (PRBN)	ESPRIT	Sequential Build-up	3%
Hierarchical Progressive Coding (HPC)*	University of Hanover	Progressive Build-up > 4 stages	3%
DPCM Using Adaptive Binary Arithmetic (ABAC)	IBM	Adaptivity	3%
		Compatibility	3%
Generalised Block Truncation Coding (GBTC)	NIS	Error Propagation	3%
		Reversibility	3%
Component Vector Quantisation (CVQ)	NIS	Special Features	3%
Block Truncation Coding With Quad tree Extension (BTCQ)*	Siemens		

* denotes technique withdrawn before initial selection process

Figure 10, Techniques Registered As Candidates For International Standardisation

Figure 11, International Standards Parameters And Their Weightings

The initial selection process at Copenhagen consisted of two parts - technical evaluation and subjective testing. Each item considered in the selection process was assigned a weighting. Weightings were originally proposed by the Japanese NIS group and modified by JPEG. Subjective quality was the most important parameter and was assigned 50% of the marks. Complexity was considered the next most significant parameter and was allocated 25%, and the remaining 25% was divided between the other second order parameters as shown in figure 11.

The CCITT NIC group has produced sets of weightings for a range of different applications including videotex, teleconferencing, facsimile and an ideal NIC service. The JPEG weightings are closest to NIC's ISDN videotex weightings, but to allow for wider applications several parameters are included, such as sequential build-up and compatibility, not required by videotex.

The technical features were judged by the JPEG technical experts following a technical presentation of each technique. Each parameter on the evaluation scoring sheet (see figure 11) was specifically addressed. Presentation was kept strictly to 15minutes to give all proposers an equal opportunity.

KTAS's image processing equipment the purchase of which had been partially funded from the ESPRIT budget was used for the subjective testing. This consists of a VAX 11/730 mini-computer connected to a VTE studio quality frame store with a two-picture capacity. KTAS have a subjective testing room with the correct surrounds and lighting conditions but the restrictive timescales did not allow the normal subjective testing procedure with neutral laymen testers. Instead it was decided that pictures would be judged by all 28 technical experts. To do this five Barco studio monitors were provided at the back of the conference room and about 5 or 6 experts sat around each.

For the initial stage (0.25 bit/pixel), all the pictures were first quickly viewed in order to gain an overall impression of the quality, then each picture was viewed for approximately 30 seconds in a random order. This single stimulus test was scored on a single linear scale marked bad to excellent. Excellent represented a very good quality with only a slight loss of resolution. Also all contents should be visible and there should no obtrusive artefacts present.

It was decided to judge the main stage at 0.75 rather than 1.0 bit/pixel as the 0.75 results produced were considered acceptable for many applications and the higher compression provided better discrimination between techniques. The main stage and the final stage were judged using a double stimulus method where for each compressed picture the original and compressed picture were alternately viewed a number of times (6 times for 5s). The viewer was not aware of whether the first or second picture of a pair was the original and had to score both pictures. Viewers marked the pictures on pairs of linear scales. Excellent represented a quality good enough for most applications with no noticeable artefacts.

The subjective testing score sheets were read into a computer via a camera input system and the technical evaluation marks were entered manually. Results were produced for each delegate showing the scores he/she had given to each technique and the total average score for each parameter for all techniques was calculated and shown in figure 12.

		Transform Techniques					Predictive Techniques			Other Techniques	
		DCTVQ	ADCT	ATADEC	LBBD	BLT	PCS	PRBN	ABAC	GBTC	CVQ
Subjective Quality	1st stage	5.34	4.45	2.30	2.36	4.14	5.01	3.19	7.17	2.86	5.48
	2nd stage	7.35	8.26	6.52	6.15	4.32	5.08	6.08	5.34	6.51	6.00
	3rd stage	9.60	9.97	6.09	9.03	6.76	9.92	9.91	9.68	9.41	9.91
	sub-total	7.17	<u>7.67</u>	5.63	5.48	4.53	5.55	<u>5.99</u>	6.14	6.07	6.29
Complexity sub-total		5.28	<u>6.92</u>	7.04	6.02	7.90	7.20	<u>7.12</u>	6.54	6.80	6.96
Other Features sub-total		5.24	<u>5.24</u>	5.61	4.89	5.90	7.44	<u>6.14</u>	6.84	6.24	5.07
Overall total		6.29	<u>6.94</u>	6.00	5.02	5.72	6.40	<u>6.32</u>	6.40	6.29	6.19

Figure 12, Results Of The ISO/CCITT JPEG Initial Selection Process

The subjective testing for the 0.75 pictures was repeated on another day and very similar results were obtained to the first test. The 1.0bit/pixel pictures were also tested later and the results showed the same techniques performed well. There repeatability gives the results further validity even though not all the normal subjective testing procedure could be followed.

5.2. Results Of The Initial Selection

The subjective testing proved very successful, most delegates were unable to recognise their own or other techniques. Hence the testing was blind and everyone considered the results genuine. The technical evaluation from the presentations, however, was less satisfactory. It was difficult for experts to assess the technical features of all the techniques equally well and it was sometimes difficult to mark objectively. Consequently it was decided to base the initial selection more on the quality than on the feature scores.

The technique with the best overall score and by far the best subjective quality marks was the ESPRIT ADCT. It was the main stage on which this technique scored particularly well. All four test pictures were of very high quality and only just distinguishable from the original. The ESPRIT results are now being verified and CCETT is to lead an international group, including DEC (USA) and NEC (Japan), to further develop the technique for the final selection meeting in January '88.

Of the predictive techniques PRBN produced the best main stage results but IBMs ABAC (Adaptive Binary Arithmetic Coding) technique produced outstanding pictures at the initial 0.25bit/pixel stage. Ideas from the PRBN technique are to be incorporated in the second group that has been set up to develop a predictive technique to be lead by IBM.

A third group is to develop a technique which may either be based on the Japanese GBTC (generalised Block Truncation Coding) or component VQ (Vector Quantisation).

5.3 The Final Selection Process

The results of the leading transform, predictive and 'other' techniques are now being independently verified. ESPRIT's ADCT is being checked by NEC and BT is checking IBM's ABAC scheme. The results on tape are being checked against those obtained by the verifier from the executable code.

For the final selection meeting technical features are to be judged from live demonstrations on prototype terminals. These will be used to assess the subjective quality of the progressive and sequential build-up. Also decoder complexity will be judged from the demonstrated decoder and from supplied cost estimates.

Subjective quality will again be judged by technical experts at three stages with compressions of 0.25, 0.75 and 3.0 bits/pixel. Additional marks will be awarded to techniques providing an additional low bit stage <0.125 bit/pixel and a final identical picture at 8 bit/pixel. An extended set of test pictures will be used which will take account of other applications and encoding will be performed as part of the test procedure.

6. CONCLUSION

During the last year of the project the consortium has concentrated its efforts on developing a transform and predictive technique for submission to ISO. As a result of intensive effort very good results have been demonstrated and ESPRIT's two techniques are being incorporated in two of the three techniques being developed for the final stage of the ISO selection process.

The original technical requirements of the project for the compression technique have been surpassed. The consortium's two techniques, ADCT and PRBN, produce very good quality results at a compression of 0.75bit/pixel. These techniques give a progressive build-up and are capable of real time decoding at 64kbit/s. The ADCT technique produces excellent quality main stage pictures and provides some compatibility with moving picture coding. The PRBN technique offers the possibility of real time decoding in software and an extended progressive build-up.

The ESPRIT consortium is now fully cooperating with the international standards community, participating in the verification of results and in the final development of the techniques. Recent work plans have been modified in response to the international standards requirements and schedule. There is a tremendous motivation and cooperation in these international development groups and already some further enhancements have been made.

ESPRIT is cooperating in the development of the two techniques that came top at the initial selection

stage and so there is optimism that the consortium will be part of the team developing the eventual standard. Because of its excellent technical facilities provided by the ESPRIT project the final selection process like the initial selection process will be held on our home ground at KTAS. The test pictures will be provided by the IBA.

The ESPRIT project has given an impetus to the international standards-making process and will provide the European partners with the knowhow and a lead in its implementation.

It is believed that further development of photographic techniques should be aimed at specific applications and should include potential users in any project. Possibilities include the development of an international network of terminals with distributed picture databases, capture/display terminals for picture transfer and the development of combined still and moving picture terminals. An extension of the project might also be considered for testing the techniques for other services such as facsimile.

ACKNOWLEDGEMENTS

This paper is based on the work of all members of the PICA consortium as listed in figure 2 and in particular the sections on PRBN and ADCT are based on contributions from Dennis Tricker (BT) and Birger Niss and Joergen Vaaben (KTAS) respectively.

Acknowledgement is made to the Director of Research of British Telecom for permission to publish this paper.

REFERENCES

- [1] Hudson, G.P., PICA - Photovideotex Image Compression Algorithms, ESPRIT'86: Results And Achievements, Elsevier Science Publishers B.V. (North Holland), 1987
- [2] CEPT Recommendation T/CD 6.1, Videotex Presentation Layer Data Syntax, Montpellier, 1984
- [3] Hudson, G.P., The European ESPRIT 563 Project, ISO TC97/SC2/WG8 N266, Boston, March 1986
- [4] Yasuda, H., The Situation On Photographic Compression Techniques In Japan, ISO TC97/SC2/WG8 N374, Parsippany, November 1986
- [5] Sebestyen, I., Study Of New Forms Of Image Formation, Communication, Storage And Presentation, ISO TC97/SC2/WG8 N392, Parsippany, November 1986
- [6] Patent Application PCT/GB86/00060, Image Coding (The RBN Principles), 1985
- [7] Patent Application, Encoding Images (Triangular RBN), 1986
- [8] Vivian, R.H., As DPCM Approaches Middle Age, Can It Keep Up With The Younger Generation? PCS '87, Stockholm
- [9] Leger, A., Duhamel, J.P., Sicre, J.L., Madec, G., Knoeffli, J.M., Distributed Arithmetic Implementation Of The DCT For Real Time Photovideotex On ISDN, PCS '87, Stockholm

Project No. 1057

**MIAC - ESPRIT PROJECT 1057 FOR
MULTIPOINT INTERACTIVE AUDIOVISUAL COMMUNICATION**

W J Clark

Video Systems Section
British Telecom Research Labs
Martlesham Heath
Ipswich IP5 7RE
United Kingdom

1 INTRODUCTION

In the future a large proportion of communication terminals are likely to be multifacility with audio as a permanent requirement. There has been considerable effort in obtaining standardisation on each facility in its own right: for example facsimile in CCITT SG VIII and audio in SG XVIII; however before this project started, there had been relatively little work on how these various facilities could be combined in a way that does not conflict with the emerging international recommendations, but will allow the implementation of compatible multifacility terminals.

Within CEPT there was an initiative in the TR1 group (now named NA3) to produce a Frame Structure for use at 64 kbit/s for audioconferencing and for second generation videoconferencing at nx384 kbit/s, that addressed the problem of multiple services within one digital transmission path. This information is now contained in draft CCITT Recommendation Y221 (see COM XV-R 16E, page 152, November 1986).

A sub-group of NA3 known as HP6 had been considering the protocols necessary to carry out multifacility audio point-to-point and multipoint conferencing and it became clear that many aspects of these Framing Structures and Protocols needed practical testing and demonstration, both to investigate their validity and to identify the various human factor and system aspects. Accordingly an initial submission was made to the CEC by BTRL, CNET, CSELT, DNL, FACE, STL, TRT * for a collaborative ESPRIT Project to prove these ideas and demonstrate the possibilities for Multipoint Interactive Audiovisual Communication (MIAC).

2 PROJECT PROPOSAL

In March 1985, a first proposal was made for a three-year Project whose aim was:- "To develop and demonstrate a system for the simultaneous communication of speech, visual and data forms of information between persons at two, three or more widely spaced locations. The demonstration itself would be a multipoint international audioconference system with visual and office type aids, but the signal and protocol infrastructure developed would be applicable to a wide range of other audiovisual services."

Following a meeting with the CEC in July 1985, at which CTNE * joined the list of partners, a new Proposal was submitted for a 27 month Project along similar lines. This proposal, which included a Demonstration system of reduced dimensions was accepted by the CEC.

Following this agreement the Project started officially on January 6 1986. Before this, a first Project Co-ordination Committee (PCC) Meeting was held in November 1985 to discuss the Contract document, the Collaborative Agreement, and the Project Definition phase.

* See Section 12, Abbreviations

3 SUMMARY OF MIAC PROJECT

The Project is first and foremost the development of a system for improved communications for meetings between groups of people. Users are likely to be in more than two different places, necessitating a fully interactive multipoint configuration. While speech alone serves many communication needs there are many others where the simultaneous transmission of visual information (images, text, indications, graphics, and so on) can greatly enhance the effectiveness of the discussion or negotiations. One of the requirements for the success of such services is that they must be economical, both in terminal and transmission cost. This may be achieved by developing a basic Audio Conference Terminal (ACT) to which existing standard peripherals, known in this application as Meeting Aids, may be attached to suit particular applications.

A single 64kbit/s channel will serve well for most multifacility services where moving pictures are not required. Extension to higher multiples of 64kbit/s for videoconference and videotelephony services is also envisaged.

The human factors of multipoint communication are not well known, and a part of the MIAC project is to study these aspects.

Based on the above concepts, the Project has the following essentials:

1. The development of a signal infrastructure suitable for integrated speech/image/data communication over 64kbit/s in long distance multipoint (and point-to-point) configuration, with extensibility to higher bit rates;
2. The application of this infrastructure to international multipoint audioconferencing.

4 SYSTEM DESCRIPTION

The objective of the MIAC System is to provide via 64kbit/s links a demonstration of the means for real-time communications between single users or groups of users in different locations. The AudioConference Terminal (ACT) will combine a good audio performance (7kHz bandwidth) with auxiliary facilities including still pictures, facsimile, data terminal and telewriting. In addition, for the control and enhancement of the Conference each Terminal will be provided with a Control and Indications Unit. Such a terminal provides the means for point-to-point audioconferencing.

By the use of a Multipoint Control Unit (MCU) which is located as a node in the 64kbit/s Network, it is possible to provide this service between 4 locations simultaneously. By using two MCUs in tandem, the final demonstration System will consist of 8 ACTs interconnected with 2 MCUs and is shown in Fig. 1.

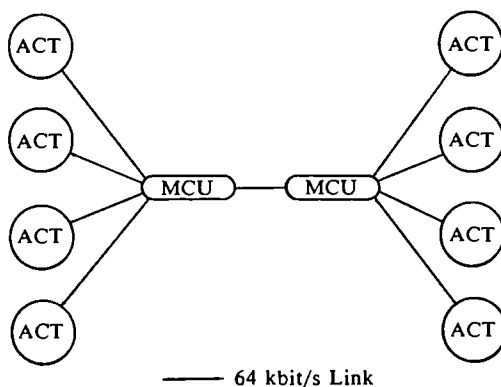


Fig. 1: ACT AND MCU INTERCONNECTIONS

4.1 The Audio Conference Terminal

The ACT provides facilities for high quality speech transmission together with the possibility of transmission from a number of Meeting Aids. These Meeting Aids include Still Picture TV Equipment, Group 3 Facsimile, Personal Computer/Data Terminal and a Telewriter. Suitable echo controllers are incorporated to permit the use of microphones and loudspeakers in an open audio system.

Each participant is provided with his own microphone. By using the Message Channel, established as part of the 64kbit/s link between terminals, each receiving terminal has an indication of the present speaker. This information appears on a 'Control and Indications Unit' or on a separate display. As well as speaker identification, the unit also allows participants to the Conference to have up-to-date status information, and permits them to control the equipment in various ways.

Each Meeting Aid is connected to an Interface and Protocol Adaptor (IPA) within the ACT. The purpose of the IPA is to convert signals and protocols from a Meeting Aid into a form suitable for transmission over the MIAC system.

A general block diagram of an ACT is given in Fig. 2.

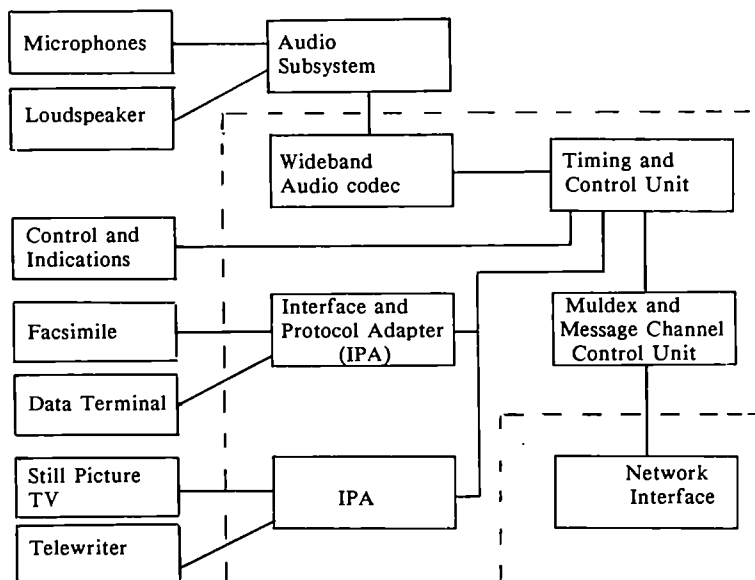


Fig 2: BLOCK DIAGRAM OF A MIAC AUDIOCONFERENCE TERMINAL (ACT)

4.2 The Multipoint Control Unit

Whilst one could envisage that a digital network which handles real time audiovisual services could provide multipoint as an integral part of its bearer service aspects, such a network seems a long way off in practical terms. The philosophy adopted in MIAC is to treat the multipoint call as a number of bidirectional point-to-point calls to a special multipoint conference unit which in the simplest case forms a centre of a star network. This therefore relieves the general switching network from the burden of continual reconfiguration within the fixed channels.

An MCU is used when more than two ACTs are to be connected together. The MCU offers the facility of voice mixing for a number of inputs, and in addition provides the means for meeting aid information and messages to be passed between a number of ACTs. A block diagram is shown in Fig. 3.

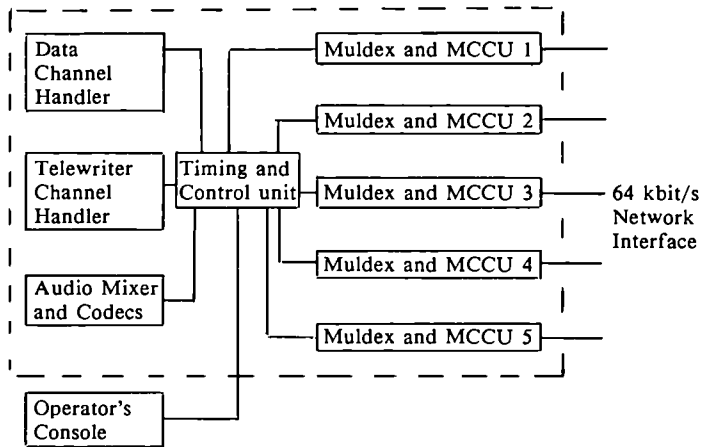


Fig 3: BLOCK DIAGRAM OF A MIAC MULTIPOINT CONTROL UNIT (MCU)

The principle behind the design of the MCU is that of distributed control: messages are sent by the ACTs, giving information on the functions required to be performed by the MCU.

5 INFRASTRUCTURE

The infrastructure of the multipoint system is based on the frame structure and message channel systems developed in CEPT/NA. A significant contribution of the MIAC project is the derivation of the precise messages and protocols to allow the use of wideband audio coding and the various meeting aids in the practical multifacility system.

5.1 Framing Structure

The frame structure is based on the idea of subdividing the 64kbit/s channel, as given in Draft CCITT Rec Y221 (COM XV-R 16E, page 152, November 1986).

This channel, which might be called the "audio-plus channel", is available as 8 bit bytes, 7 of which can be treated as individual sub-channels of 8kbit/s while the 8th is designated as a service sub-channel. An audio channel is formed from a block of sub-channels giving a net bit rate of $n \times 8\text{kbit/s}$ with $n = 1-7$. This audio channel uses the CCITT Rec G722 speech algorithm capable of working adaptively at 56 or 48 kbit/s conveying speech at a higher quality than telephony. Sub-channels that are not used for audio are available for Meeting Aid data. The 8kbit/s service sub-channel provides synchronous information to describe the use of the remaining 7 sub-channels as well as a transmission capacity for very low speed information such as telewriter information. There is also a 4kbit/s message channel using layered protocols.

The service sub-channel, as well as indicating the use of the remaining 8 bits within the 64kbit/s or 8 bit byte, can also signal the use of other channels external to the 64kbit/s. In this way a common frame structure can be used for audio and videoconferencing. Fig. 4 illustrates the frame structure implemented.

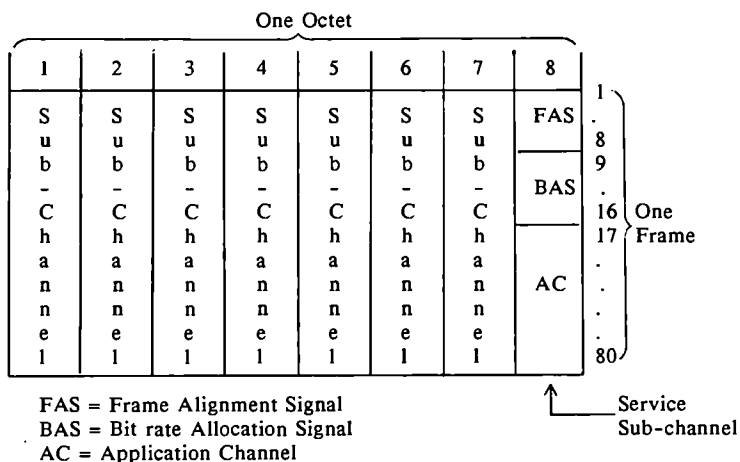


Fig 4: Y221 Frame Structure

The Bit rate Allocation Signal (BAS) allows the transmission of codewords to describe the structure of the residual 56kbit/s channel as well as the structure of the primary rate multiplex in which the 64 kbit/s channel may be inserted. The latter only applies in the case of $n \times 64$ or $m \times 384$ kbit/s services such as videoconference or videotelephony.

Sixteen frames are structured as a multiframe, with two 8-frame submultiframes. The BAS is repeated 8 times along the same sub-multiframe and a majority 5 out of 8 decision is used to provide error resilience. The validated BAS then applies to the next sub-multiframe. A change in configuration can therefore occur at sub-multiframe rate, ie, after 80ms, and a data channel can be inserted or removed without interrupting the audio channel.

The application channel (AC) in bits 17-80 of each frame provides a user bit rate of 6400bit/s. As each bit in a frame represents a bit rate of 100 bit/s, any synchronous channel working at $n \times 100$ bits can be inserted in this application channel. It is proposed for teleconferencing that the Application Channel will be used to provide an 800 bit/s Telewriter Channel and a 4kbit/s Message Channel. For future extension it is also possible to insert data channels at the bit rates in the hierarchy defined in CCITT Recommendation X1 and when needed forward error correction or encryption information could also be carried.

The system thus provides a simple economic, well proven, but flexible framing philosophy, very similar to that used in the 2Mbit/s videoconference codec described in CCITT Rec H 120/130. It allows the control of a higher multiplex configuration into which the basic 64kbit/s is inserted, which implies that MIAC will be inherently compatible with any future generation video telephone or videoconference system which adopts such a structure.

5.2 Protocols

In the MIAC project a number of protocols are being studied and refined. These relate to the main areas of conference management (for example chairman control of voice channels) and the management and use of the Meeting Aids.

Within a Teleconference, five separate phases can be distinguished. These are: Reservation, Set up, Session, Recovery and Reconfiguration, Disconnection and Call Release. Within the conference session, the equipment must make provision for the control of the voice, and for the correct operation of the Meeting Aids.

5.2.1 Voice Management

Face-to-face conferences can take a number of forms. They might be very formally structured, with participants having to gain permission to speak from a chairman, or at the other extreme they could be small, informal, meetings in which all participants have equal status.

For teleconferencing, these two types of conference are referred to as "Conducted" or "Non-Conducted", respectively.

For a conducted conference, the Chairman may speak and be heard by other participants at any time. If however a participant wishes to make a contribution he must first send a "Request for the Floor" message to the Chairman. As in a face to face meeting, the Chairman may permit or disallow the contribution. Thus the Chairman retains full control over the participants.

For a non-conducted conference, no chairman is necessary and the sound from each ACT is sent to all others. In this case it is left to the participants to moderate their speech so that a meaningful discussion can be held.

A third type of Conference known as "half-conducted" has also been provided for in the MIAC system. Here the MCU takes the form of an automatic chairman, permitting up to three simultaneous speakers.

5.2.2 Meeting Aid Protocol

One of two approaches can be adopted towards the protocols to be used for Meeting Aids in a multipoint environment.

The first approach is to permit each Meeting Aid to have direct access to the 8kbit/s data channel, and allow the unadapted signals to flow from one ACT to another. Whilst this approach is feasible for point-to-point operation it means that the MCU must contain special facilities for each type of Meeting Aid. In particular this approach would make it difficult to add extra facilities since any new Meeting Aid would require a specific facility within the MCU.

A second approach which has been adopted within the ACT is to interpose an Interface and Protocol Adaptor (IPA) between every Meeting Aid and its access to the data channel. The IPA converts the particular data format from the Meeting Aid into the basic format used within the MIAC system which will allow the MCU to broadcast the data transparently to each terminal.

Thus from each ACT a point-to-point data link is established with the MCU. Within the MCU, data received from one ACT is broadcast to all other ACTs.

If any information needs to be sent back to the transmitter from a receiving meeting Aid, then this and any negotiation required will be performed over the Message Channel.

Using this approach, the same design of MCU may be used with any type of Meeting Aid provided the ACTs involved in the conference have compatible IPAs. In the future, Meeting Aids using X25 layer 2, such as Group 4 facsimile, can be accommodated by an additional data handler in the MCU.

5.2.3 Message Channel

For the control and enhancement of the Audioconference, a 4kbit/s Message Channel is established on a point-to-point basis over the Application Channel between ACTs (in the case of point-to-point) or each ACT and its associated MCU.

The message channel is structured in accordance with the concepts of the ISO OSI 7-layer model.

Layers 1-6 of the protocol on the message channel are in accordance with the CEPT NA3/HP4 specifications, while the layer 7 was specified taking into consideration the particular type of facilities required during an audioconference. A list of 20 facilities was identified related to different phases of the audioconference.

The application layer protocol on the message channel was conceived for the realization of the facilities and makes use of a set of about 60 messages. The messages themselves are encoded according to CCITT Rec X409.

Each message within a facility sequence consists of the following blocks:- Function Code, Destination Address, Source Address, Optional Information Field.

These messages are routed using information contained within the layer 3 protocol. Layer 2 of the protocol ensures the correct error-free reception of the messages.

Messages may be generated either in response to actions of the Participants, or by the equipment itself. In both cases the exchange of messages is not directly visible to the users since the Control and Indications Unit, mentioned in Section 6.1 converts when necessary the inter-ACT messages into a convenient form understandable to the participants.

Much of the work of the Project is in defining, implementing and testing the structure of this control message system.

6 MEETING AIDS

6.1 Control and Indications Unit

This unit is realised as a software package which can be run on a personal computer. The function of the C&I unit is twofold: To enable the chairman, if appointed, to control the progress of the Meeting, and to allow participants at each Terminal to have some knowledge of the status of the Conference.

A mouse-driven screen is provided for ease of use which gives the name and location of the present speaker, request-for-the-floor indications, and other menu orientated access to Terminal facilities and System information.

6.2 Telewriter

The Telewriter has been developed as part of the MIAC Project, and enables the exchange of handwritten information such as text drawings and diagrams. The input device used consists of a writing tablet together with an attached pen, with most of the surface of the tablet available as the writing area. The remainder is reserved for a number of soft keys, which perform such functions as line width selection or erasure. The output is displayed on a colour TV monitor.

In order to avoid confusing the user with a number of screens, the SPTV and Telewriter facilities have been combined into one unit, and share a single display. When first switched on, the display indicates to the user that the equipment is in the telewriting mode. The marker is switched on and is visible if the pen is moved within 2.5 cm of the tablet surface.

Three types of image are possible: one where the telewriting image is shown alone; another where it is superimposed on an SPTV image; and a third where the marker can be moved over both the telewriting and SPTV images.

6.3 Still Picture Television

The SPTV equipment has also been specially developed as part of the Project. It is connected to a standard CCIR 625 line colour camera and monitor.

In use, when the user selects the SPTV mode, the screen can either clear to black or recall a previously transmitted picture. A command line appears at the bottom of the screen which contains all the options available, selectable by a mouse.

In the first prototype these are kept to a minimum set of: reversion to Telewriter; view the source video in real time; capture the incoming video in the picture store; send the picture or cancel.

Whilst the final version of this device will make use of the results of the ESPRIT 563 Picture coding algorithm for photographic videotex when available, the flexible hardware used for SPTV can implement alternative algorithms in the meantime. Using a typical algorithm, picture update times of between 50 seconds and 3 minutes can be achieved. For transmission, the data is blocked in HDLC format with the block size dependent upon the coding method used.

6.4 Facsimile

One major feature of the MIAC system is that standard group 3 facsimile equipment is used for this service. For connection to the ACT the facsimile meeting aid operates as if connected to a leased line using a two-wire analogue connection.

Binary coded signalling procedures are used according to CCITT T.30 with CCITT V.21 channel 2 modulation in synchronous mode at a data rate of 300 bits/s as described in T.30. Facsimile data is transmitted using CCITT V.29 modulation at a data rate of either 7200 or 4800 bits/s according to the procedures defined in T.30 and T.4.

In operation, the user can operate the facsimile in a simple way. He has merely to insert the document to be transmitted into the machine and press the send button. All signalling is carried out automatically by the MIAC system. Another important feature is that documents can be transmitted simultaneously to as many locations as are currently connected, unlike normal facsimile "broadcasts" which have to transmit to each location one after the other.

6.5 Data Terminal

File transfer between personal computers is an important feature of any office system. However, when the original project duration was reduced by 9 months, it was decided that this part of the work should be included only if time permitted. As a result, although it is hoped to demonstrate this aspect in a simple way, a more detailed study and rigorous protocol definition would ideally form part of a subsequent project.

7 PROJECT ACTIVITY

7.1 Project Management and division of work

At the first Meeting of the Partners, it was decided that the work of the Project should be divided into two main parts: that of overall co-ordination would be managed by the PCC (with representation from each Partner) which would meet at least 3 times per year, whilst the more detailed technical work of the Projects would be handled by five smaller specialist working parties (WPs).

Each WP comprises relevant personnel from the Partners and has a Convenor who is responsible for the day-to-day running of his WP. Each month during the study phase of the Project the Convenors were also responsible for reporting progress to the PCC. The WP topics are listed below in Sections 7.2-7.6.

At the first Meeting of the PCC, it was agreed that the Working Parties 3, 4 and 5 could not effectively start work until the Project was more closely specified, and should wait until a Draft Functional Specification had been written. When this document had been agreed, each of the WPs held meetings, both by audioconference and by travel. For the day to day matters, the EUROKOM system has been extensively used, and found to be a quick and convenient way of communicating between diverse locations.

During this time each WP produced Specifications for its particular equipments, and these were agreed by the PCC. Although the WPs work collaboratively, each Partner is individually responsible for the deliverable of his Company. Workplans were produced showing the exact effort and time to be applied to each task of the Project, in terms of both WP and Partner.

7.2 WP1 Audio Codec

This WP has as its objective the provision of hardware and software to implement the Wideband speech coding algorithm standardised by CCITT SGXVIII. Interfaces between the ACT and Audio-subsystem were defined and suitable equipment practices agreed together with a common DSP device. By suitable design an audio codec card common to both MCU and ACT is provided.

7.3 WP2 Audio Subsystem

This WP was responsible for the design of the audio parts of the ACT. This part of the equipment is responsible for picking up the sound, amplifying and making echo control processing, digitizing the signal and interfacing to the Audio Codec. This included making the necessary physical arrangements of microphones and loudspeakers, provision of simple signalling facilities for use by participants for Floor Request, and the option of individual displays of active talkers.

The main part of the work involved the specification and development of the Analogue Interface, the Power amplifier, the speaker control and the D/A and A/D boards.

7.4 WP3 Audioconference Terminal

This WP had the task of defining the complete ACT including the various Meeting Aids. Based on the use of a VME rack, interfaces between individual cards were defined which made possible the parallel development of the various parts of the ACT at the different locations of the various Partners. Of course this approach leads to some increase in the component cost of the prototype ACT, but this is to some extent offset by the convenience of development and equipment and software interchange.

As well as the ACT, this WP was responsible for the design of the Meeting Aids which will be used in conjunction with the audio path. These devices are connected to the ACT by means of Interface and Protocol adaptors which interface to the VME rack.

7.5 WP4 Multipoint Control Unit

This WP was responsible for the design and implementation of the Multipoint Control Unit. This device makes possible the interconnection of a number of individual ACTs offering audio mixing together with the means for handling control messages and data flows from the various interactive Meeting aids. By discussions with other WPs it has been possible to use a number of common cards in the ACT and MCU.

7.6 WP5 Protocols, Messages and Framing Structures

This WP was charged with the task of providing the protocol infrastructure upon which the rest of the MIAC Project would be based. A number of protocol specifications were written, concerned with the facilities to be offered by the MIAC System and the way in which these might be practically implemented. Necessary control messages were defined and methods proposed for their encoding in accordance with International standards. The information thus provided has formed a firm basis for the work of the other WPs, and an interactive process has taken place to further detail and refine the specifications.

Following the extensive Project definition and specification which was carried out previously, the WPs 1-4 were in a good position to start the more practical work of hardware and software development.

8 HUMAN FACTORS ASPECTS

The Human Factors aspects could be summarised as the two topics of "Ease of use" and "Usefulness of the Service". Within the remaining time of the Project it is intended to address these topics by various means.

As a result of tests and experimentation by the end of the Project there will be a System with various options which is easy to use, and a Report on the Usefulness of the service is planned.

There are a number of areas for study: Overall Room design and layout; Speech quality and intelligibility; Conference Management; Usefulness and application of Meeting Aids; Training and Instruction for users.

8.1 Room Layout

Fig 5 shows a typical equipment layout. This room plan was based on the full MIAC implementation of Meeting Aids and participant positions. Whilst BTRL will be building a room based on this design, other Partners will be able to provide alternative facilities for test purposes. To avoid any criticism that MIAC could be too costly and complex, some tests will be carried out with a smaller simpler ACT with perhaps just one microphone and a single Meeting Aid, operating in an Office type environment.

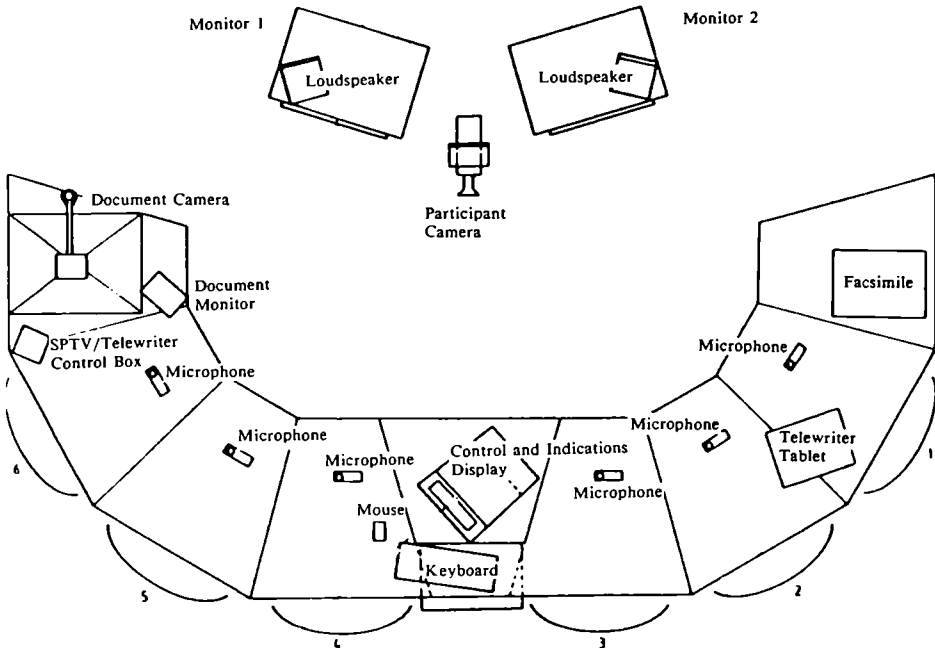


Fig 5: POSSIBLE CONFERENCE ROOM LAYOUT

8.2 Speech Quality and Intelligibility

A number of subjective tests have already been carried out by CCITT SGXVIII. The purpose of the tests are not to repeat this work, but rather to ensure that the methods of audio alignment, echo control etc. provide for maximum performance in a range of conference rooms.

8.3 Conference Management

Under this topic comes the study of both speech management and Meeting Aid Control. The central factor here is the use which can be made of the Control & Indications Unit. This will be studied from both the point of view of convenience of use, and whether the facilities provided for conference management are adequate and necessary. In particular the role of the Conductor and the benefits or otherwise of the various conference modes has to be verified.

8.4 Usefulness and Application of Meeting Aids

Having established that each Meeting Aid operates correctly in a Multipoint environment, some study will be made of the benefits of the different meeting aids to various types of conference. The Human Factors aspects of using such facilities as telewriting in conjunction with SPTV can also be evaluated as well as some consideration of the best way of displaying a number of different items of information within the same conference environment.

8.5 Training and Instruction for users

Ideally, the equipment should be so simple and self explanatory, that no formal training in its use is required. Even so, previous tests of teleconference equipment have shown that various factors are at work when users are confronted with new terminals. First, the users need to be given a clear conceptual model of the equipment which they are expected to use; for instance the telewriter may be described as being like an electronic blackboard. Second, users may suffer from fear of technology if the equipment presentation appears to them too complex to operate readily. In addition, the equipment should have some similarity to equipment with which users are familiar. To guide the user, simple diagrams may be of help and these approaches will be developed towards the end of the Project.

9 RESULTS ACHIEVED

From the start of the Project, excellent co-operation has been a feature of the work. Most of the first year of the Project was used in the detailed specification and early development aspects. Towards the end of 1986 the first prototype muldex and message channel cards were produced which were the first implementation of the Y221 framing structure.

In November a successful demonstration was given of two audio subsystems working back to back in Turin. This showed the high fidelity of the audio performance.

Audio codecs using the G722 algorithm were produced and tested, and early in 1987 two completed ACTs were tested, firstly locally in a laboratory and then in a point-to-point connection between two locations in the UK. At this time, facsimile, the first Meeting aid was introduced. Results obtained from these tests showed the potential that could be realised from such equipments, in convenience of use and benefits to users. For those who have not experienced this means of communication but are more familiar with audioconferences using the PSTN, the audio improvement may be likened to the difference between listening to radio reception on VHF/FM and that obtained on MF/AM. In addition, the enhanced status information available eases the work of the participants, particularly that of the Chairman. The correct operation of the equipment also gave confidence in the protocol specifications which had been implemented.

Development of the MCU has reached the stage of the first tests with an ACT, and by September 1987 it is intended to be able to operate three ACTs in a multipoint configuration, each having a single Meeting Aid.

During this time standardisation information was also provided from MIAC into the relevant international standards bodies.

10 CONCLUSIONS

ESPRIT I057 has made possible a collaboration of diverse organisations, who have been prepared to tailor their personal and national interests for the benefit of the Project as a whole. From this working together, a communications system is being developed which promises to show the way for future audiovisual services.

It is apparent that the advent of digital circuits will make possible the provision of a wide range of audiovisual services as well as other multifacility services not involving audio. There is an opportunity, before these services proliferate and their transmission formats diverge, to standardise a set of protocols, with a suitable and flexible multiplexing scheme whereby the various facilities may be carried in any combination on a single call. This contribution has described the start that has been made within the MIAC project to achieve this aim; however much work still needs to be done.

11 ACKNOWLEDGEMENTS

The author wishes to acknowledge the spirit of co-operation and the work carried out between his colleagues both at BTRL and within the other partners to the Project (CNET, CSELT, DNL, FACE, STL, CTNE, TRT*). Thanks are also due to the Director of Research and Technology, BTRL for permission to publish this paper.

12 ABBREVIATIONS

CNET- Centre National D'Etudes des Telecommunications-France
CSELT- Centro Studi E Laboratori Telecomunicazioni SpA - Italy
CTNE - Telefonica -Spain
DNL-Dr Neher Laboratories -Holland
FACE- Industrie FACE Standard SpA- Italy
STL -STC Technology Ltd -UK
TRT -Telecommunications Radioelectriques et Telephoniques -France

Project No. 925

A EUROPEAN PROJECT ON CODING OF MOVING AND STILL PICTURES AT VERY LOW BIT RATES

J. DAVID - J.P. HAAG ALCATEL- CIT

The ESPRIT Project 925 is dedicated to low bit rate videoconferencing coding algorithms. The main objectives of this project are :

- to achieve very high compression rates down to allow operation on 64 - n X 64 Kbit/s networks including the emerging ISDN. The compression rates considered are 0.1 - 0.05 bits/pel for moving pictures with good resolution and quality and 0.2 - 0.5 bits/pel for still pictures with high resolution and quality.
- to promote practical standards with industrial support strong enough to promote these standards as de facto solutions and to ensure products compatibility in order to create the basis for a real development of the market.

1. INTRODUCTION

Inside the ESPRIT Program, our six companies, ALCATEL-CIT (France) -prime partner, GEC (U. K.), PKI (FRG), SAT (France), SEPA (Italy) and TELEFONICA (Spain), which have good experience in both freeze frame image and videoconferencing systems, consider that it is mandatory, for future technological development to create a level of European standardization in image coding as well established as it is now with COST Project at 2 Mbit/s.

The project is limited initially to algorithm research and standards specification and it is divided in four workpackages :

- Workpackage 1 : State of the art in image transmission and videocompression.
- Workpackage 2 : Survey of technical requirements in teleconferencing.

- Workpackage 3 : Technical specifications for moving image codecs in the range 384 -64 Kbit/s.
- Workpackage 4 : Technical specifications for still picture codecs in the 64 Kbit/s range.

The eighteen month project started in January 1986 and has resources of 18,75 man years.

2 - STATE OF THE ART IN IMAGE TRANSMISSION AND VIDEOCONFERENCING

A review of the technology to date was carried out to establish a reference point for all future work. The survey covered a review of available products and known developments in the low bit rate image coding of moving and still pictures and a review of literature to establish research and achievements.

CURRENT AVAILABLE PRODUCT

The following products available or announced have been considered.

- AVELEX	U. S. A.	19.2 - 448 Kbit/s
- WIDCOM	U. S. A.	56 Kbit/s
- PICTEL	U. S. A.	56 - 128 Kbit/s
- COMPR. LABS	U. S. A. Rembrandt 56	56 - 384 Kbit/s
- V. T. T.	Finland	48 - 64 Kbit/s
- NEC	Japan Netec X D	56 - 112 Kbit/s
- MITSUBISHI	Japan	56 - 1544 Kbit/s

Widcom, Avelex and Pictel solutions are the most widely marketed in that field. CLI Rembrandt 56 has been just introduced in early 87. The introduction of the NEC product has been delayed several times and the other products seen fairly confidential.

Evaluation videotapes of three products (Widcom, Avelex and Pictel) have been reviewed. The PICTEL product ranks first for picture quality.

The reviewed products use various videoprocessing techniques, albeit DCT based systems are the most widely used. It is not possible from this product survey to make any definitive assumption on a potential best solution regarding performance or ease of implementation.

PATENTS

A search for patents has been carried out mainly by interrogation of data bases ; cosine transform coding methods has been patented by several companies or organisations.

STANDARDS

As basis for the video codec simulation, review of existing standards and prestandards considerations has been performed . The most relevant activities are those of CCITT SG XV Specialist group OKUBO on coding for visual telephony, which works on 384 - n x 384 Kbit/s coding based on transform coding techniques. The partners of the project are also in this workgroup.

LITERATURE SURVEY

The following subjects have been examined :

- Pre and Post filtering
- Transform coding
- Hybrid coding
- Motion compensation and detection
- Still picture
- Other coding techniques
- Quality evaluation metrics

CONCLUSION

In conclusion, a cosine transform type with motion compensation seems to be a good choice to obtain a high compression efficiency with a good quality for moving video.

3 - SURVEY OF TECHNICAL REQUIREMENTS IN TELECONFERENCING

A review of major operations requirements associated with videoconferencing has been carried out, with special attention at :

- Video terminal considerations
- Networking aspects of image codecs

- International connections and compatibility

VIDEO TERMINAL CONSIDERATIONS

The videoconferencing terminal shall be dependant on the application
Three main application classes can be considered :

1 - Multiperson-codec is a codec which must be able to transmit all the video, audio and auxiliary information needed for a conference between two groups of participants. Video equipment must provide suitable means of framing the scene, slides, documents and other visual facilities and of displaying them in such a way that all participants should be able to follow them without been distracting from the conference.

2 - A personal codec for office environment is used by a single person at each site and must provide, besides audio and video connection, some auxiliary functions to exchange documents, graphics and pictures.

3 - Home codecs may be thought of like a telephone with video transmission facility that can be used by the costumers like a normal telephone, with built-in camera and screen.

It is considered that 384 kbit/s may be used for first and second classes and 64-128 kbit/s may be used for second and third classes; provision must be taken for audio bit rate; telephone quality (0.3 - 3.4 KHz) and wideband audio (0.05 - 7 KHz) can be considered in the range 16 to 64 Kbit/s.

NETWORKING ASPECTS OF IMAGE CODECS

Currently non switched networks at 2 048 Kbit/s and 1 544 Kbit/s are available. Switched 64 Kbit/s ISDN is under introduction. A single frame structure has been defined for use on these networks :

- For moving picture the video bit rate should be considered in the range of 248 - 320 Kbit/s for 384 Kbit/s and the range 48 - 96 Kbit/s for the range 64 - 144 Kbit/s.

- The frame structure has to take into account operation on bit rates as low as 64 Kbit/s, which dictates subframing of 64 Kbit/s channel : it is proposed to use the new 64 - n x 64 Kbit/s frame structure Y 221 currently standardized in CCITT which allows : frame alignment signal, bit rate allocation signal in the network, application channel.

For delay considerations, the time delay introduced by video processing has to be kept to minimum especially if a satellite link is involved on the transmission path. A critical parameter for the codec delay is the buffer memory size and the video format conversions. An estimation of codec delay give 250 ms. An objective of maximum delay is 300 ms. The buffer management should be efficient to avoid the use of long buffer delay.

Taking into account the previous experience with the current CODEC COST 211, an important point is to use optionally forward error correction.

Maintenance and supervision of equipment connected to a public network must allow separable actions for two main parts :

- Maintenance and supervision of the subscriber line, which is related only to the network and the interface with the used equipment.
- Maintenance and supervision of the subscriber equipment itself.

The facility of multipoint connection is one of the objectives in videoconferencing design since this facility allows a number of codecs to operate in conference mode, as opposed to point to point connection. It has been decided to solve this problem in the same way as for 2 Mbit/s videoconferencing, by using a central MCU (Multipoint Connection Unit) with the method of fast update request capability.

The number of conferences has been examined with respect to the network layout ; in particular if the number exceeds five (typical figure) more than one MCU is required. If the network is asynchronous, the MCU must impose its own clock on to all the codecs.

Other important points on multipoint are interaction and compatibility between codecs with different options or different standards, encryption and compatibility with other classes of services e.g. audioconferencing...

INTERNATIONAL CONNECTIONS AND COMPATIBILITY

Codec compatibility : The problem may occur between codecs working with different options or standards, e. g. video standard NTSC or PAL/SECAM. Regarding this last aspect, the already available standard for an intermediate video format for 384 kbit/s operation, mandatory for international and also national use, shall solve the main compatibility problem. Care shall be taken in final specification to ensure compatibility of the other options of the codec (graphics, audio, data, etc...).

Network compatibility : The problem occurs in constitution of international "digital pipes", for example between 2 048 and 1 544 Kbit/s networks or 64 - 56 Kbit/s networks. It may be solved by the use of network converters, which already exist for some applications including COST 211, CCITT H 110 - 120 - 130 videoconferencing.

4 - TECHNICAL SPECIFICATION FOR MOVING IMAGE CODECS IN THE RANGE 384 - 64 Kbit/s

Work has been especially focused on the development and simulation of low bit rate image coding algorithm in the 64 Kbit/s range because important progresses had been carried out by the CCITT at 384 Kbit/s. Analysis and comparison (all simulations are carried out on the same sequence of images so that the relative merits to the different methods can be more easily assessed) allow to select a final solution and to derive technical specifications and proposals for implementation.

PRELIMINARY REQUIREMENTS

As a result of previous considerations, the coding algorithm must comply with the following requirements :

- Conversion of video inputs or outputs to an intermediate format suitable for straightforward video standards conversion (e. g. PAL - NTSC). Coding and decoding processes are performed on this internal video intermediate format independantly of any conversion process from a particular video standard (e. g. PAL - NTSC) to intermediate format. Coding considerations dictate the use of the CCITT Okubo Group intermediate format (CIF : 30 Hz, 352 pixels by 288 lines for luminance, 176 pixels by 144 lines for chrominances) with possibly modified resolution for 64 - 384 Kbit/s operation.
- Operation on CCITT time slot structured digital channels from 64 to 384 Kbit/s ; at 384 Kbit/s 5 time slots of 64 Kbit/s shall be used for video and another time slot shall convey audio and miscellaneous ; operation down to 192 Kbit/s shall be performed by reducing the number of video time slots ; operation at 64-128 Kbit/s is to be structured after better insight on video data rate reduction at this level ; provision is taken for automatic control of the digital channel allocation (dynamic data allocation, frame alignment). In all cases the frame Y 221 currently defined on the CCITT will be used.

-Possible multipoint operation using a network Multipoint Channel Unit, implying some protocol handling for multipoint operation control.

TECHNOLOGY CONSTRAINTS

Practical hardware considerations will effect the possible implementation of a codec and hence that are to be taken into account in the development of algorithms and technical specifications.

- Market restrictions : future high quality codecs used in videoconferencing rooms must not exceed price and physical size of current generation codecs independant of the increased processing complexity ; future lower quality codecs operating at 64 Kbit/s should be priced and sized significantly below current generation codecs.

- Technology limitations : it is forecasted that the algorithms suitable to achieve the necessary coding efficiency and quality are bound to involve some complex image processing functions which shall not be easy to implement in hardware and also will not be executed in real time in software on usual processors. Along with the continous technology progress on usual devices (RAMs, EPROMs), new candidate devices will possibly be used for efficient implementation ; these candidates are for example :

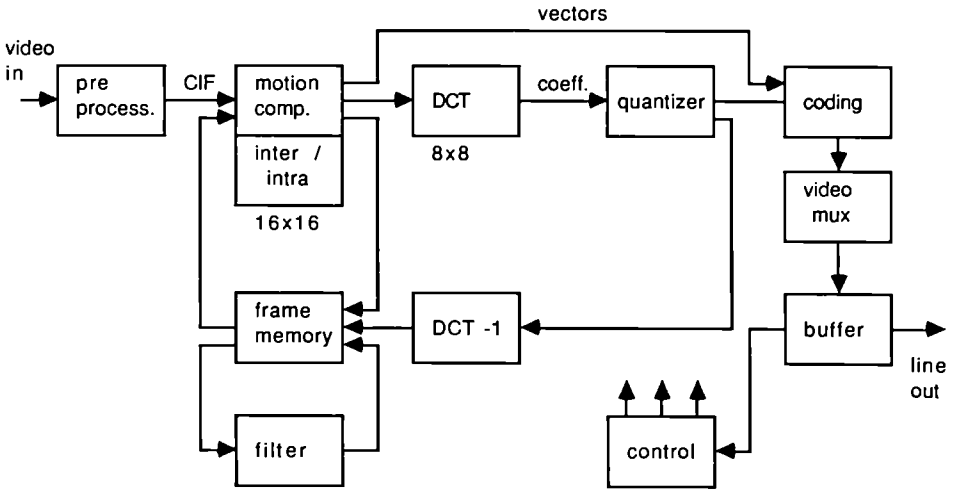
- . digital signal processors
- . parallel operation processors
- . semi-custom hardware devices (e.g. standard cells)

The resulting algorithm considerations are as follows :

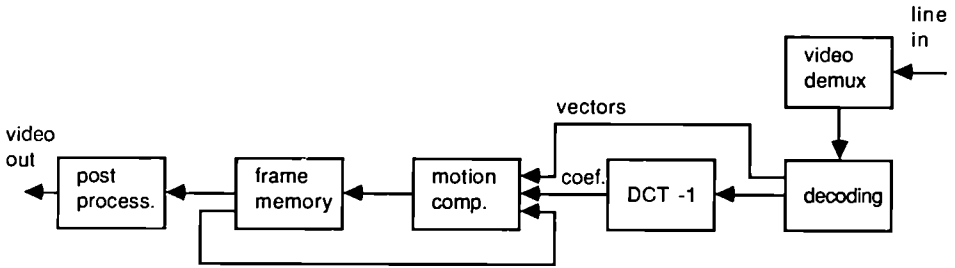
- keep RAM arrays to reasonable sizes, especially in inter-frame processings
- limit as many as possible true multiplications.
- favor pipe-lined implementations
- avoid as many as possible recursive processings
- limit where possible arithmetical precision

REFERENCE MODEL

A basic architecture and coding-decoding processes based on motion compensated transform coding have been specified as a "reference model"
.(fig 1)



CODER



DECODER

fig 1 : Codec Architecture

This reference model :

- takes into account an intermediate format with the following characteristics :
 - . 352 pels by 288 lines luminance (CIF)
 - . 88 pels by 72 lines chrominance (by 2 reduction of CIF)
 - . basic frame frequency of 10 Hz
- is based mainly on motion compensated transform coding.

For coding considerations, this reference model is not identical to the CCITT reference model selected for 384 n x 384 Kbit/s operation and contained two alternatives for 64 - 384 Kbit/s operation :

- . motion compensated DCT on 8 x 8 blocks : this source coding method is seen as most probable standard for 384 - n x 384 Kbit/s operation ; this alternative aimed at easy compatibility of source coding between 64 - 384 and 384 - n x 384 Kbit/s operations.
- . motion compensated DCT on 16 x 16 blocks : this alternative aimed at achieving maximum source coding efficiency for operation at very low data rate, whatever coding method is standardized at 384 - n x 384 Kbit/s rates.

VIDEO CODING ALGORITHM ANALYSIS

Simulations have been carried out starting from the reference model based on motion compensated DCT 8 X 8 or 16 x 16 blocks applied to CCITT SG XV test sequences (Miss America, Trevor, Claire)

First evaluations shows that:

- the 16 X 16 blocksize is more effective regarding picture quality - but it is less compatible with n x 384 Kbit/s.
- the color resolution 90 X 72 is too low and it is preferable to work with the standard CIF 360 X 288 for Y and 180 X 144 for Cr and Cb.

Different possibilities have been investigated to obtain enhancements; the main ones are the following :

- different law of quantization depending on the type of block (in motion , intra-coded ...)

- concept of macro block 16 X 16 with DCT 8 X 8
- Limitation of number of zero coefficients
- Post processing
- 2 D runlength coding
- fullsearch for motion estimation
- loop filter on motion block only
- DC components coding in Intraframe mode by DPCM
- Reduction of the resolution of video format

The best enhancement is the concept of macro-block 16 x 16 using a DCT 8 x 8. A Y macro block is a group of 4 Y 8 x 8 blocks forming a 16 x 16 block and a C macroblock is a group of 2 Cr 8 x 8 blocks and 2 Cb 8 x 8 blocks.

This concept allows :

- 1 - compared to 8 x 8 DCT, to decrease the number of bits dedicated for the overhead (the choice inter-intra and the motion vector are calculated on the macroblock) while preserving a coding efficiency comparable to 16 x 16 DCT
- 2 - to use a DCT 8 x 8 (DCT 16 x 16 is more complex)
- 3 - to obtain therefore a compatibility with scheme n x 384 Kbit/s

A post-processing applied on the whole picture also allows to obtain a better subjective quality.

We have also introduced a reduced common intermediate format 256 pixels by 192 lines for Y and 128 pixels by 96 lines for Cr and Cb. The quality of the pictures are very close and are still under evaluation ; choice might depend on application considered.

FINAL SPECIFICATION

The partners propose a specification of Hybrid coding at 64 Kbit/s. This coder support common intermediate format, a max frequency of 10 Hz and a blocksize of 8 X 8 for the DCT.

The Hybrid coder is specified as follows :

- Pre-processing Y = 352 pixels by 288 lines
 Cr and Cb = 176 pixels by 144 lines
 frequency max = 10 Hz.

 Possibility of reduced format
 Y = 256 pixels by 192 lines
 Cr and Cb = 128 pixels by 96 lines
 frequency max = 10 Hz.
- Motion Search length = +/- 7
 Compensation block size = macro-block 16 X 16
 a single motion vector is generated for a group of
 4 Y blocks 8 X 8
- Coding DCT size = 8 X 8
- Inter - "a priori" on macroblock 16 x 16.
 Intra Decision After a scene cut, only intra mode is used during
 the whole picture. The first picture is coded
 with half rate (skipping the second frame at 10
 Hz).
- Quantizer To reduce the number of bits in intraframe mode
 and avoid a buffer overflow after a scene cut,
 parts of the DC coefficients of an intra-frame
 coded macro block are transmitted in form of
 their difference to preceding DC coefficients.
 The quantizer step size is evaluated after each Y
 row and C row.
- Scanning diagonal only
- Video A group of blocks consists in 22 Y macroblocks
 multiplex 16 X 16 followed by 11 C macroblocks
 arrangement
- Filtering A 2 dimensional 1. 2. 1. filter within a 16 X 16
 macroblock is used after the frame memory if the
 motion vector is unequal to zero.
- Buffer The transmission buffer size is 8 Kbits.
 strategy The quantizer stepsize g is chosen as a function
 of buffer content.

IMPLEMENTATION ISSUES

The proposed standard can be implemented in various ways. Apart from a VLSI and/or semi-custom strategy implementing on silicon the main operators (DCT, motion compensation, 2D filters ...) , alternate designs based on parallel arrays of commercial processors might also constitute viable solutions.

5 TECHNICAL SPECIFICATION FOR STILL PICTURE CODECS UP TO 64 Kbit/s

The work on still picture coding has consisted of study and simulation of techniques for multilevel (several luminance levels) and binary still picture coding , considering that:

- coding of photographic still pictures can be performed by either Photographic Videotext coding or by using moving video coding techniques in still picture mode,

- transmission of binary or multilevel images is of importance for teleconferencing applications.

The main effort has been dedicated to the derivation of quality evaluation metrics for video images, including moving video; quality measurements are well defined for audio signals but have been up to now very limited (e.g. signal to noise ratio) for video signals.

CODING TECHNIQUES FOR STILL PICTURES AND GRAPHICS

The studies made for still picture coding can be divided into two great fields:

- multilevel grey images
- binary images

A review for still picture coding standards and other works carried out on several related projects on this subject has been made. The research and simulation work have been developed in the light of their conclusions.

The work on multilevel images has been centered on the use of Vector

Quantization applied to DCT as a definitive option. Previously, other algorithms have been tested, such as those based on ADPCM , hierarchical encoding, visual perception models, direct vector quantization, etc , including some hybrid methods combining vector quantization with other algorithms, mainly hierarchical ones.

The work on binary images has gone from vectorisation (transformation into graphics), passing through some hybrid methods combining two-dimensionnal features grouping points of the images , down to typical run lenght algorithms. The vectorization needs a very specific sort of images, and therefore , in spite of the best results obtained using this technique, the final recommandation would be an intermediate algorithm, on the sense of less compression but more generality in the images wich can be processed.

QUALITY EVALUATION METRICS

The aim of this work has been to establish and define a possible set of standard quality evaluation metrics, based on subjective criteria.

The first step has been the creation of the set of questions to be made to the observers, with the objective of avoiding redundancy and lacks. Some statistical n-dimensionnal processes have been applied to a great quantity of data over a test sequence, until a final set of standard questions and answers can be stabilized.

In a second step, the different algorithms investigated for moving video coding have been evaluated using the results of the first step. Results have been presented and used in order to compare the proposed algorithms and to derive the final standard solution for a European generation of videocodecs.

6 - CONCLUSIONS

The ESPRIT 925 project aims at achieving quality moving and still picture transmission at $64 - n \times 64$ Kbit/s.

The results have allowed to give a proposal of standards of a potential application for face to face videotelephone at 64 Kbit/s. The scheme of hybrid coding at 64 Kbit/s is proposed under constraints of compatibility with 384 Kbit/s.

The CODEC is fairly complex but the partners are confident that the algorithms can be implemented with acceptable hardware costs.

Future work should be considered toward CCITT standardization by the mean of prototype developments.

7 - ACKNOWLEDGEMENTS

This paper written by ALCATEL is based on contributions from the partners of ESPRIT 925 : ALCATEL, GEC, PKI, SAT, SEPA, and TELEFONICA.

Project No. 64

TEXT-TO-SPEECH SYNTHESIS IN AN OFFICE ENVIRONMENT

D. BOILLON (*), H. BUETHER (+), J.P. LEFEVRE (&),
L. NEBBIA (!), L.C.W. POLS (#)

- | | |
|--|---|
| <p>(*) D. BOILLON
S.E.S.A.
Rennes Atalante
3, rue du Clos Courtel
BP 1897
F - 35018 RENNES CEDEX
Tel. (33) 99.63.50.50</p> | <p>(!) L. NEBBIA
CSELT
Via G. Reiss Romoli, 274
I - 10148 TORINO
Tel. (39) 11.21.69.417</p> |
| <p>(+) H. BUETHER
NIXDORF-NME
Berliner Strasse 66
D - 1000 BERLIN 27
Tel. (49) 30.43.85.366</p> | <p>(#) L.C.W. POLS
University of Amsterdam
Institute of Phonetics Science
Herengracht 338
NL - 1016 CG AMSTERDAM
Tel. (31) 20.525.21.83</p> |
| <p>(&) J.P. LEFEVRE
OROS
Chemin des Prés - Zirst
F - 38240 MEYLAN
Tel. (33) 76.90.62.36</p> | |

1. INTRODUCTION

The prime objectives of the ESPRIT-SPIN Project (P64) "Speech Interface at Office Workstation" are to achieve significant advances in the different speech areas (coding, recognition, speaker verification and text-to-speech synthesis) in order to integrate them in a comprehensive speech interface for an office environment workstation. Such objectives, when achieved, will allow to develop competitive future workstations using speech as an input/output medium.

The use of speech techniques such as recognition, coding, or even speaker verification, is generally considered to be useful in an office environment. However, with respect to text-to-speech synthesis opinions are more diverted.

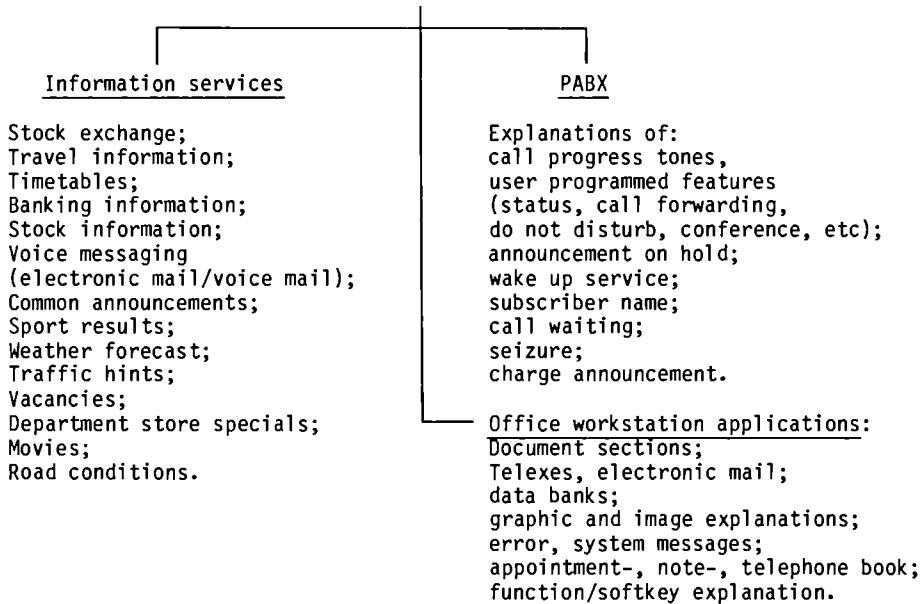
So the aim of the present paper is first of all to establish clearly in an office environment those applications for which text-to-speech synthesis is useful and coded speech cannot be used and secondly to highlight some of the major developments carried out on text-to-speech synthesis in the framework of the SPIN Project.

2. ADVANTAGES OF TEXT-TO-SPEECH SYNTHESIS IN AN OFFICE ENVIRONMENT

The integration of text-to-speech modules in the Computer Integrated Office (CIO) can be related to the following application scenarios:

- (voice) information services with access over the public and/or private telephone network;
- office workstation general applications;
- PABX specific information and announcements.

TEXT-TO-SPEECH APPLICATIONS AT THE CIO



- Information services

Most of today's applications using text-to-speech are controlled through the telephone. Because of the restriction to only two acoustic channels, voice has to be used for the transmission of information to the user, whereas voice or touch tones can be used for command entry. Examples for applications are all kinds of message and information systems for which it is important to reach them from every telephone (weather forecast, timetables, stocks, banking, data banks). This market is still advancing.

Most of the information stored in these systems change very rapidly. But there are also certain kinds of information, such as the time of day, which are highly constrained both in vocabulary and syntax, and which can be "synthesized" by simple concatenation of pre-recorded words and phrases. Recorded and concatenated speech allows straightforward access to the stored information. More complex retrieval operations, such as searching or scanning the data base are not feasible with recorded voice. Many existing data base systems allow extremely sophisticated search and retrieval of information. Text-to-voice synthesis technology makes these data base operations available through the telephone. A special application is the voice messaging area.

Normally user messages and system prompts are stored as coded voice. But sometimes it is necessary to include messages in a user mailbox, which are originally in written form. So an operator needs to speak the messages into the mailbox. Whenever it is necessary to store a lot of messages, a text-to-speech system will ease this task. Additionally, generation and editing of system prompts could be done in a flexible and economic way.

- Office workstation

. Desk top applications

Text-to-speech in the office environment is not seen today in many actual applications. Nevertheless there are useful applications. Examples are reading out of document sections, explanation of graphics and images, telexes, electronic mail, and the content of data banks. The verbal output of such sections could be of great help when the user is working with another task and needs this information.

An other example is an electronic dictionary which could help when reading or preparing a document in a foreign language.

Annotations to text, graphics, and images are normally recorded with coded speech. But perhaps a user would prefer writing his remarks instead of speaking them, because written text can be more easily edited than coded speech.

. Messages

For error and system messages, coded speech is already used. Most of these messages are very simple. So coded phrases or the concatenation of coded words is used.

For more sophisticated messages, supplied with additional hints (which has to be done by an operator), the concatenation of coded speech sounds phony, because the super-segmental parameters of the word-forms are inconsistent. The intelligibility cannot always satisfy all needs. Text-to-speech offers more possibilities to adapt the voice to the importance of a message by altering the sound of a voice, the stress of the sentence, the speed and the loudness.

. User guidance

A wide area for text-to-speech is the user guidance for features of a PABX, for workstations, machines and computer aided instructions. Computer-aided instructions means that someone studies or works on something (machines, electronic components) and gets information and instructions by a computer (e.g. theory of operation, functional descriptions, remove and replace instructions, repair procedures, and trouble-shooting procedures). Using text-to-speech instead of coded speech makes it easy to change or to complete the information without having the necessity to re-record the spoken parts. It remains necessary to have access to the written text. The user also has the ability to browse through the written information when searching for a particular part. This is much easier than to browse through spoken information.

Another point, and the most important for us, is the user guidance in applications. For voice messaging systems and PABXs the only "terminal" is the telephone. All the messages used have to be recorded with a professional speaker when using coded speech. Every change in the structure of the system or the adoption for foreign countries makes it necessary to re-record parts of, or the complete message, which is an expensive and time-consuming procedure.

Using prerecorded messages makes it nearly impossible for the user to change any of them. Changing the messages is more and more needed in today's sophisticated applications, which allows the creation of the interfaces by the users themselves. Changing the structure of the interfaces makes it also necessary to change the hints and messages. When using text-to-speech, changing the spoken messages is as easy as changing the displayed messages.

All these considerations led us to emphasize text-to-speech synthesis studies in the SPIN Project, always keeping in mind to build, as far as possible, language-independent tools. An overview of the main results obtained until now is given in the next section.

3. WORK DONE IN THE FRAMEWORK OF THE SPIN PROJECT

The text-to-speech synthesis systems developed in the SPIN project are based on a segmental representation approach for both acoustic and prosodic aspects. Let us briefly describe the various models involved in this system.

- Data base of elementary segments

The idea of using segments as minimal speech elements comes from difficulties in finding adequate rules to model spectral variation in natural speech. In our systems, we use basically LPC encoded diphone elements, an ideal diphone being defined as the region lying from the middle of one phone to the middle of the subsequent one. With this definition, coarticulation influence, which usually does not extend much further than half a phone, is taken into account without specific rules.

To avoid restrictions to the above definition, it appears that the diphone data base must contain nearly all possible pairings of phonemes for the language under consideration, even if a lot of these pairs only occur at word junctions. Also, the possibility of context dependence must be avoided in the generated set, because such dependencies which are specific for each language would restrict the overall objective of transparency.

Temporal and spectral characteristics of the diphone templates must be as closed as possible to occurrences of these diphones in natural speech. To achieve these requirements, diphones are extracted from real speech. In order to ensure the possibility of using one unique diphone template in a wide range of contexts, it appears that the words in which the diphone templates are embedded must be phonetically neutral. Practically, this eliminates the potential utilisation of real words where articulatory targets are often undershot. So, we have designed some procedures in order to generate automatically, from the list of phonemes, a set of nonsense words containing all the possible diphones of the language under consideration.

A single native speaker is used to record the complete set of previously defined nonsense words. This task is accomplished quickly and under good conditions with an interactive software facility. After recording, the words are analyzed on a frame by frame basis.

Nevertheless, the main part of the effort in the dictionary compilation is the accurate segmentation of the speech waveforms. The semi-automatic process includes two major steps. First, an automatic diphone localization is performed. Then, a second step of semi-automatic user verification is added, in order to achieve the desired boundary detection accuracy and to ensure the coherence of the data base. This verification process involves interactive displays of the relevant parameters and automatic segmentation results, and allows for some possibilities of subjective auditive test. It is obvious that best results are obtained when the above control procedure is supervised by an expert of the language under consideration.

After verification, the satisfactory segment contents are added to the data base. In addition to the classical acoustic parameters (LPC representation), the format of the base allows the introduction of some supplementary information, which is of interest for the subsequent stages, like voiced/unvoiced indicator, consonant micromelody, limits of the phone transitions, and so on.

- Morphologic and syntactic analyses

The main aim of the morphologic and syntactic analyses is the decomposition of the sentence in groups of words and the attribution of a unique assignment, in terms of phrase-level unit, to each of these groups, referred to as sense groups. Moreover, these analyses contribute to the removal of ambiguities, apt to occur during the orthographic-to-phonetic transcription process.

To perform these complex linguistic analyses, a specific rules compiler called SCYLA has been developed in the framework of the SPIN Project. More details about SCYLA will be given in section 3.2.

The results of the syntactic analysis, combined with outputs of the morphologic analysis, provide for each identified sense group some specific prosodic markers. These markers reflect the syntactic environment of the sense group, from which will be inferred the corresponding prosodic structure. The rules governing the elaboration of these markers may be changed, in accordance with the language.

- Orthographic to phonetic transcription

This processor translates an orthographic input into a sequence of phonemes, in accordance with the phoneme inventory. This involves a set of context-dependent rules able to determine the exact correspondence between a grapheme sequence and its phonetic counterpart based upon phonetic features of adjacent letters.

The size of the set varies to a great extent depending upon the considered language and its pronunciation difficulties. For example, English or Italian are complicated by the need to predict the stressed syllable location, whereas French words have always stress on the last syllable. Also in Italian, rules are needed to discriminate between hiatus and diphthong in vowels sequences like ia, ua, and so on. On the other hand, French has a specific complexity related to liaisons which may occur over word boundaries. Undoubtedly, such

examples could be extended. Anyhow, the knowledge of morphologic and syntactic analysis results allows simplifications in the rule description as well as better overall performance.

- Prosody generation

Each language has its own prosodic structure, which can be discovered only by extensive investigations carried out on a large amount of natural speech, in order to extract the evolution of prosodic factors like fundamental frequency or duration. The main problem, when considering the multilanguage aspect, is how to manage these "prosodic rules", resulting from the above analysis, which are definitively language dependent.

The basic idea we have retained is to build a set of prosodic patterns (intensity, melody, rhythm) to be associated with each sense group. Such patterns will be accessed by parameters related to:

- . the syntactic environment (through the prosodic markers);
- . the phonetic context (like number of syllables, nature of these syllables, and so on),

then directly and easily fitted on the acoustic string without any language restriction.

In the beginning, this technique follows an approach similar to the use of diphones as acoustical elements. Now, we try to isolate the minimal "prosodic segments", without deeply understanding what their content is. With this approach the language characteristics are taken into account when elaborating the pattern access parameters; but they are transparent for the synthesis software since these characteristics are confined inside the patterns. It is obvious that the pertinent phonetic context parameters can be different depending upon the language under consideration. Also the relative importance of each sense group, as well as the sense group segmentation itself, has to be performed by language specific rules, as has been mentioned above. The access parameters can be viewed here as the counterpart of the phoneme inventory associated with the elaboration of the diphone data base.

In contrast, elaboration and compilation of the prosodic patterns, extracted from natural sentences, are performed basically as language transparent processes, with the help of semi-automatic procedures comparable to those involved during the acoustic segment data base generation phase.

More details about the studies carried out in the prosody area will be given in section 3.1.

- Segment concatenation and synthesis

Speech synthesis from a diphone data base is now a well-known technique. The only required care is to check the continuity at segment boundaries. We ensure this continuity by performing amplitude and spectral smoothing, limited to the steady state portions.

Then, the prosody treatment is performed by superposition of the appropriate prosodic pattern on each sense group. This superposition is made by scaling in time the segments from the dictionary in order to obtain the same duration between the prosodic pattern and the concatenated diphone string. Almost linear temporal deformation techniques, anchored on syllable nuclei, have been

implemented in order to ensure this function. The deformation has to be applied only on the steady state portion, in order to preserve the natural timing of the transitions. It could be noted that the superposition process cannot use classical time warping, due to the different nature of the two entities (distance calculation should have a sense). Finally, fundamental frequency is assigned to each frame by direct transfer from the corresponding frame of the prosodic pattern. In addition, and when required, consonant micromelody information, extracted from the diphone dictionary, can be added. Ultimately, pauses are introduced in the string.

After that overall presentation of SPIN text-to-speech synthesis systems, our goal is now to present in more detail the work done in three specific topics:

- determination of prosodic rules of intonation and duration for the French and Italian languages;
- the SCYLA rules compiler;
- the evaluation of the speech quality of rule synthesis for French.

3.1. Determination of prosodic rules of intonation and duration for the French and Italian languages

Prosody has been studied extensively in the SPIN project following steps a) to d):

- a) A number of "linguistic variables", which can influence the prosody, has been initially identified. First of all the syntactic structure of the sentence to be synthesized is quite important, at least to generate the correct intonation. Information carried by intonation is related to a subdivision of the sentence into its clauses and, in turn, to a subdivision of clauses into phrases. The project of the syntactic analyser is a problem in the problem and it should be dealt with separately. Incidentally, it should be remarked that several important applications of text-to-speech do not require a full syntactic analysis (for example, access to data base where information is recorded in a sort of predefined scheme, like lists of items, addresses, features, etc).

At a lower level than the syntactic one, the main linguistic variables which must be considered for prosody generation are: phonetic context of each phoneme, position inside the word and type of syllable (stressed or unstressed), position of the word inside the phrase, length of the word (number of syllables) etc.

- b) A corpus of natural sentences (separate for duration and intonation) was selected and analysed in order to investigate and quantify the influence of different variables. The same professional speaker was used for both experiments.

Five repetitions of reiterant speech (nonsense trisyllabic words embedded in a carrier sentence) was the material used for duration analysis. A lot of measurements were performed in this phase: about 100 items for vowels (single and in sequence), 37 single consonants (in intervocalic, initial and final position), 20 geminated consonants, 110 consonantal clusters (in intervocalic, initial and final position). A further experiment was performed on four repetitions of 34 natural sentences for validation of the obtained results and extension to occurrences (mainly vocalic sequences) at word boundaries. Also the influence of a consonant on the preceding unstressed vowel has been investigated.

For intonation the selected (initial) corpus was made up of three repetitions of 55 declarative sentences. The variables taken into consideration were:

- 1) length of a clause or of a constituent;
 - 2) position of the constituent in the clause;
 - 3) position of a subordinate clause relative to the main clause.
- c) Two models, for duration and intonation, were conceived according to which the knowledge acquired in the analysis phase has been expressed by formal and general rules.

The duration model assigns an intrinsic duration to each phoneme (the reference is the unstressed single vowel and the single intervocalic consonant). A number of sets of rules takes into account separately coarticulation effects (vowels sequences, at word boundaries too, and consonantal clusters), word length and stress position, position of a word in the phrase and sentence final lengthening. Each set of rules brings a modification which is expressed by a multiplicative coefficient with a superposition effect.

After a careful study, an intonational model has been adopted in which the sentence intonation is firstly described by a succession of High and Low tones assigned to specific points by a set of linguistic rules. Then this string of Highs and Lows is interpreted as target values within an envelope and, finally, the actual intonative contour is computed by transition rules between the targets.

- d) The linguistic knowledge for prosody generation has been expressed by means of SCYLA, a rule compiler developed under this project and briefly described in the next section.

3.2. The SCYLA rules compiler

SCYLA, shorthand for Speech Compiler for Your LAnguage, is a compiler of rules, expressed on a contextual basis, according to a IF THEN ELSE structure.

A SCYLA procedure is basically able to transform string elements of an input level into string elements of an output level.

Let us suppose the simple rule for Italian phonetic transcription, which states that the sequence "cc" at letter level is translated into the (long) affricate phoneme "t\$:" when it is followed by the letter "i" or "e", both stressed and unstressed; otherwise into the palatal obstruent "k:".

The corresponding SCYLA rule in a procedure where the letter level has been declared as input and the phoneme letter as output, is the following:

```

c c      →   "t$:"
          /... i,e,"^i","^e"
          →   "k:"
          ;

```

Levels and elements belonging to them are defined by the user.

A SCYLA program is made up of procedures, each of them scans the input level element by element, applies the first rule, if any, whose context is matched, and moves to the next input element until the end is reached.

Alignment between elements of different levels created by successive procedures is managed automatically by the system.

A large number of possibilities is allowed to express a context string: mixing of elements of different levels, combination of elements in OR, AND, NOT, reference to classes of elements defined by the user, etc.

A numerical (integer or floating) level can be declared (for prosodic modules) and a value in the string can be inserted or modified by an arithmetic operation.

A set of rules can be triggered by a pre-context condition, in order to accelerate the execution of the program.

Strictly speaking, SCYLA is a program translator, since it produces a "C" source program which can be compiled by any computer provided with a "C" compiler. The synthesis system can be developed on a general purpose computer, exploiting SCYLA facilities (easiness of writing the rules, clearness of documentation). Then the system can possibly be adapted to a specific application and finally can be moved to the target (microcomputer) system without rewriting any part of the software.

3.3. Evaluation of the speech quality of rule synthesis for French

In order for a text-to-speech rule-synthesis system to be useful in an office environment, such a system should be optimally integrated in the office workstation. One prerequisite for that will be a certain minimal speech quality. If the generated speech is almost unintelligible, or very unnatural, or very attention-demanding, then such a voice-output system will never be accepted by the user, not even when all other ergonomic aspects of the man-machine interface are very well taken care of.

Therefore, speech quality evaluation is a necessary component of the ESPRIT-SPIN research program. The term "speech quality" is very broad and should be subdivided and specified in order to be testable. Our interest has been mainly devoted to segmental intelligibility, in other words the intelligibility at the phoneme and syllable level. Later tests will also involve supra-segmental intelligibility, including prosodic attributes at word and sentence level. At that level it also makes sense to evaluate other aspects like naturalness, acceptability, and memory load of rule-synthesized speech.

Although several languages, French, Italian and modern Greek are involved in the speech output part of the SPIN project, the speech quality evaluation so far has been limited to French, because only for that language a full diphone set is presently available.

The French rule-synthesis system uses about 1,250 so-called diphones as its basic units. The transition between two speech sounds is part of the unit itself and therefore does not have to be described in rules, contrary to allophone-based synthesis systems. By carefully concatenating these units one can generate words and sentences.

With 34 French phonemes, about 1,250 diphones are needed for a full coverage of all possible VV, CC, VC, and CV combinations (V=vowel, C=consonant). For a detailed diagnostic evaluation of all diphones it is not sufficient to use the common type of CVC words, so we chose for CVVC- and VCCV-type nonsense words. A complete set consists of 14 lists of 50 words each. Apart from training stimuli, two complete sets were presented to two groups of four native French listeners. For comparison a subset of the test words was naturally spoken, then analyzed and resynthesized by using LPC (linear predictive coding) and also presented to the subjects. The task of the listeners was to identify the four phonemes in each test word with four key strokes.

The stimulus words were prerecorded on analog tape and presented over headphones in 4-sec intervals to four subjects simultaneously.

- Test results and conclusions

Seven subjects showed a rather consistent behaviour, whereas one subject could not keep up the pace and gave many incomplete responses or no response at all. The raw scores were lined up in such a way that incidentally missed phonemes and missed words were replaced by one question mark per phoneme. Furthermore, apparent typing errors were corrected and certain dubious responses were reinterpreted by giving the listener the benefit of the doubt. An algorithm for this correction phase was also developed.

The average results, as given below in figure 1, are based on the two actual test runs for the seven subjects. Because of the specific structure of the word material used and the difficulty of the task in general, one should not rely on absolute phoneme and word scores, but on relative scores only. For instance, the vowel score of about 65% correct in these LPC-resynthesized words is much lower than the generally reported score of 80% and above for CVC words in English and other languages. Nevertheless the intelligibility of LPC-resynthesized speech is generally considered to be of high enough a quality to be acceptable as voice output of an office workstation. The low scores found in this experiment for the four-phoneme nonsense words are related to the similarity between, as well as the size of, the French vowel set which is quite a bit larger than for English. More important in the present results is the fact that the vowel scores for the rule-synthesized CVVC words, are another 20 to 25% lower, probably indicating non-optimal VV diphones. Similarly, the consonant scores in CVVC- and VCCV-words are 15 to 20% lower for rule-synthesized stimuli than they are for LPC-resynthesized stimuli.

Confusion matrices are available for every phoneme position in the two word types in both listening conditions.

Especially initial and final /r/ is badly identified and shows room for improvement in terms of better diphones or adapted concatenation rules. The general feeling is that diphone-based rule synthesis should be able to approach a phoneme intelligibility close to that for LPC-resynthesized speech.

A similar test for a Dutch diphone-based rule-synthesis system also did not yet come up to that score, however. Our results also clearly show that the intelligibility of a phoneme is not one unique value, but is strongly dependent upon the phoneme position (initial, medial, final) and the word type. There are strong indications that test results given so far in literature for closed-response tests with CVC words represent a far too optimistic view of the quality of rule-synthesis systems.

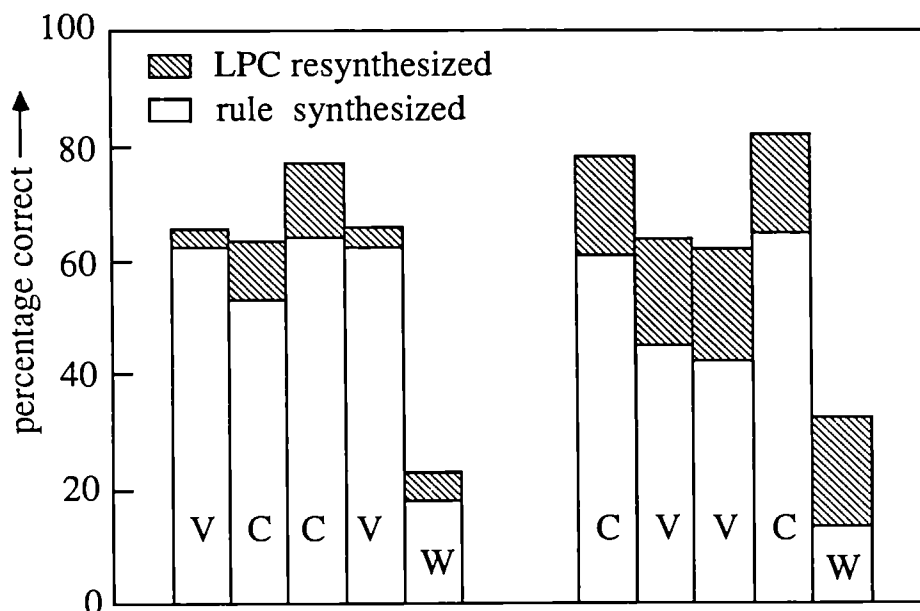


FIGURE 1

Phoneme and word intelligibility scores for CVVC- and VCCV- type test words under two conditions: LPC resynthesis and rule synthesis. The scores are averages for 7 French listeners

4. CONCLUSION

The work to be done in the next two years of the SPIN Project will lead to the building of a speech interface for an office workstation. It will include, among others, a coding board and a text-to-speech synthesis board.

Office applications are confronted with a lot of data which might be presented to the user via speech output. They will have to be precisely analysed to determine whether coded speech or text-to-speech synthesis is necessary. However, one must keep in mind that the bit rate of text is less than 100 bits/sec which is at least two orders of magnitude lower than of digital speech rate. On the other hand the quality of coded speech is much better than the present quality of speech generated by text-to-speech synthesizers. A point in favour of text-to-speech is concerned with the source of information, which source generally exists or originates in text form. The time and cost to convert such texts into voice recordings via coded speech might be prohibitive.

To conclude, let us include some considerations on the state of the market.

English (American) text-to-speech systems are already used in large systems where the costs of a text-to-speech unit are low compared to the cost of the whole system. For small systems, like office workstations, the difference in price (a coding chip costs a few dollars), is one of the important reasons which prevents the use of text-to-speech. But still the expectations for a wide breakthrough are high, as shown in figure 2 hereafter.

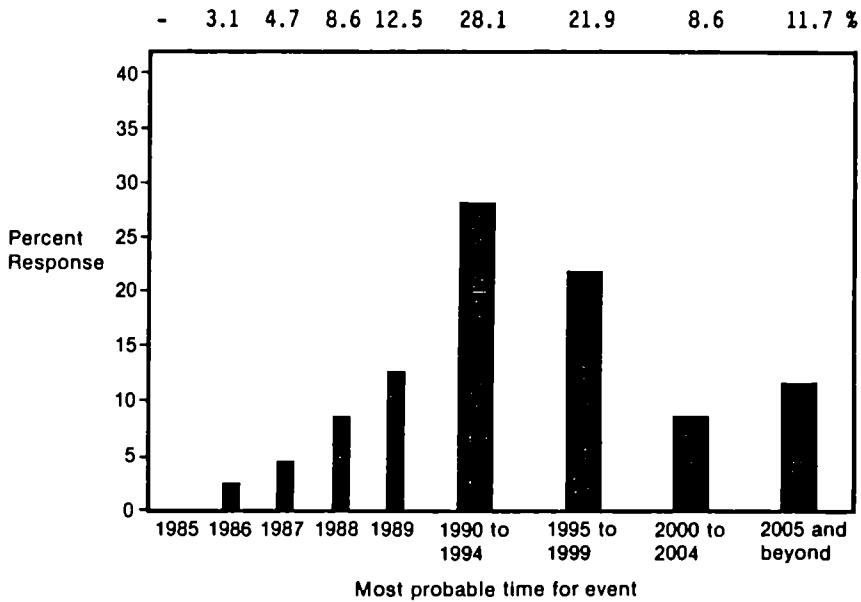


FIGURE 2
Text-to-speech event table

LITERATURE

Below we give a short list of references for some topics dealt with in this paper.

- Text-to-speech synthesis in the office environment:

G. BREIDBACH, K. FELLBAUM: Elektronische Sprachsynthese; Umsetzung von Schriftsprache in gesprochene Sprache; Technische Universität Berlin; Institut für Fernmeldetechnik.

D.Y. WONG, D.M. LINDSAY, M.S. Ng, D.Y. CHENG, J. GOLDSTEIN, R. FRANKEL: Text-to-voice synthesis in a voice application processor; Speech Tech '87; Proceedings; April 28-30, 1987, New York.

J.D. KNIFFIN: Authoring technical data for delivery by text-to-speech converters; Processor; Speech Tech '87; Proceedings; April 28-30, 1987; New York.

- Work done in the framework of the SPIN Project:

In addition to SPIN deliverables dedicated to these topics, some general papers written in the framework of the SPIN Project are included:

BEZOOIJEN, R. VAN & POLS, L.C.W. (1987)

Evaluation of two syntheses by rule systems for Dutch
Proc. European Conference on Speech Technology, Edinburgh (in press).

LEFEVRE, J.P. (1986)

A diphone speech synthesis approach applicable to different languages
Proc. IEEE-ICASSP86, Tokyo, 2443-2446.

LEFEVRE, J.P., AUBERGE, V. & MARET, D. (1987)

Alignement automatique optimal de deux chaînes phonétiques
Proc. 16th Meeting of Galf (Speech group of the Acoustic Society of France)
(in press).

POLS, L.C.W. (1987)

Quality evaluation of text-to-speech synthesis systems
ESPRIT-SAM report (in press).

POLS, L.C.W. & BOXELAAR, G.W. (1986)

Comparative evaluation of the speech quality of speech coders and
text-to-speech synthesizers
Proc. IEEE-ICASSP86, Tokyo, 901-904.

POLS, L.C.W., LEFEVRE, J.P., BOXELAAR, G.W. & SON, N. VAN (1987)

Word intelligibility of a rule synthesis system for French
Proc. European Conference on Speech Technology, Edinburgh (in press).

LAZZARETTO, S., NEBBIA, L.

SCYLA: Speech compiler for Your LAnguage
Proc. European Conference on Speech Technology, Edinburgh 2-4 Sept. 1987
(in press)

SALZA, P.L., SANDRI, S.

Microprosodic timing rules for Italian consonant clusters
Proc. IEEE/ICASSP86, Tokyo, P. 2035-2038.

AMORPHOUS SILICON CONTACT IMAGER FOR OFFICE AND GRAPHIC APPLICATIONS

N. Kniffler (MBB, BRD); J. Van Daele (Agfa-Gevaert, Belgium);
J. Nijs, Z.-M. Qian, H. Pattyn, J. Poortmans (IMEC, Belgium),
E. Fogarassy, C. Fuchs, J.C. Desoyer (CNRS, France)

The photolithographic and etching processes for the detector structure glass/TCO/pin/Al were developed. We followed two scanner concepts: a linear organized type of scanner and a matrix organized type scanner. Samples with 4 pixel/mm and 10 cm length were realized on both concepts. Photo current and dark current of the used pin structure were measured on test structures. A feasibility study for read-out, interconnection and packaging of the contact imager was done. Also the outline requirements for the imager to quantify target specs. Fundamental material studies and deposition study on homo CVD processes were done, to develop deposition techniques for higher stability of material characteristics. Also the deposition processes of a-Si:H by photodissociation of SiH_4 using UV light was studied. The basic energy transfer mechanisms, free radical chemistry and the deposition rates have been examined.

Thermal calculations have been performed to understand the recrystallization of a-Si thin films. The amorphous polycrystalline transition was characterized by X-ray, TEM and Raman spectroscopy.

An investigation of two fabrication processes has been performed showing the advantages of integration of addressing switches and scanner on the same substrate. The two processes are a high temperature and a laser recrystallization process for integrated polycrystalline TFT's together with amorphous silicon sensors.

1. INTRODUCTION

The objective of this project is on one hand to develop a linear contact imager with amorphous silicon, resulting in a very compact system and on the other hand to investigate new techniques to improve the properties of this contact imager.

In office and graphic arts applications the scanning of original documents is normally done with laser scanners or solid phase image scanners (e.g. linear CVD arrays or photodiode arrays, 2-3 cm long). In both these systems a long optical path requires considerable space. Contact imagers on the contrary can make the equipment compact and reliable. Contact imagers have the advantage that in applications with large scanning width the scanning resolution can be chosen independently from the scanning width.

A very interesting material to fabricate sensor arrays for contact imaging with high S/N ratio is amorphous silicon, because this material can be deposited uniformly over large areas.

The first two years of this project, which started January 1986 is considered as an evaluation/feasibility phase and consists of three main parts:

1 - the manufacturing of a very compact contact imager with amorphous silicon sensor elements, creating a linear scanning array. The contact imager will be packaged, incorporated and evaluated in-system. A market study has to determine the quantitative targets;

2 - investigation of alternative deposition techniques for a-Si (homoCVD, photoCVD) aiming at higher stability of the deposited films in comparison with glow discharge;

3 - study on the integration of thin film switches and shift registers on the same substrate to avoid cumbersome and expensive hybrid connections.

2. DEVELOPMENT OF AN A-SI:H LINEAR CONTACT IMAGER : TECHNOLOGY AND INTEGRATION IN A SYSTEM (BY MBB AND AGFA-GEVAERT)

The aim of the technology development is to establish photolithographic and etching processes to manufacture substrates with thin films of TCO, a-Si and Al and $30 \times 5 \text{ cm}^2$ in size.

Therefore equipment for processing substrates with that size was set up. Also measurement techniques for studying dark- and photo current characteristics of the scanners were developed.

2.1 Development of Photolithographic Techniques for Large Thin Films of TCO, a-Si and Al

After some problems of etching the SnO_2 -layer (TCO) were overcome, etching processes were established for all three layers. The roughness of an etched TCO-layer edge depends strongly on the properties of the TCO. Fig. 1 and fig. 2 are showing the patterning of a sprayed TCO and a CVD-TCO.

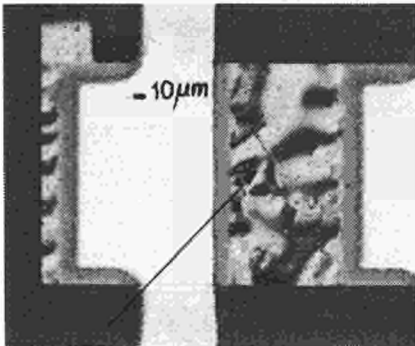


Fig. 1: Patterning of sprayed TCO

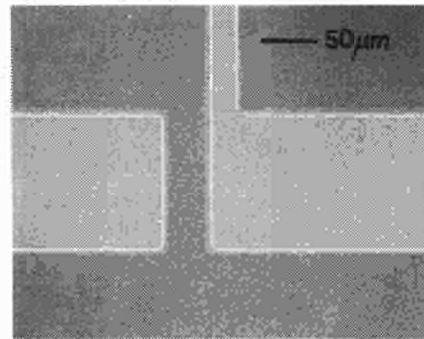


Fig. 2: Patterning of CVD-TCO

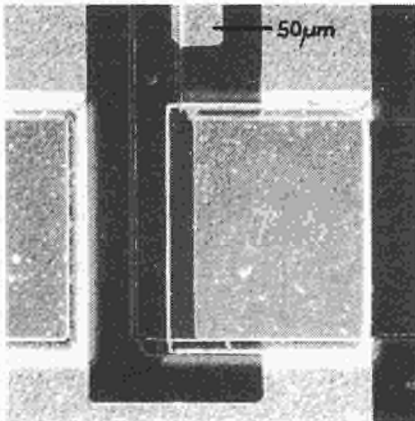


Fig. 3: Single pixel of a fabricated 10 cm/4 pixel scanner

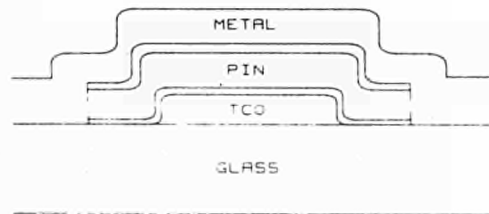


Fig. 4: Cross-sectional drawing of a single pixel (Fig. 3) showing the shortening of the pin-structure

We also managed the etching problems arising from the Si:C-p⁺-layer of the amorphous silicon pin device. The Al-layer could be etched away very easily. In fig. 3 a completely etched pixel of linescanner is shown. The scanner was processed in a very simple way: a) patterning of TCO substrate, b) deposition of pin amorphous silicon layer, c) patterning of the pin amorphous silicon layer, d) evaporation of Al-layer, e) patterning of Al-layer. The disadvantage of this very simple processing is the shortening of the p- and n-layer of the pin device by the metal layer (Al) as shown in fig. 4. The shortening of the pin device can be avoided by using a two step deposition and patterning of the p⁺- and in⁺-layer. By using this technique the first line scanner with 4 pixels/mm and 10 cm length could be realized (fig. 5).

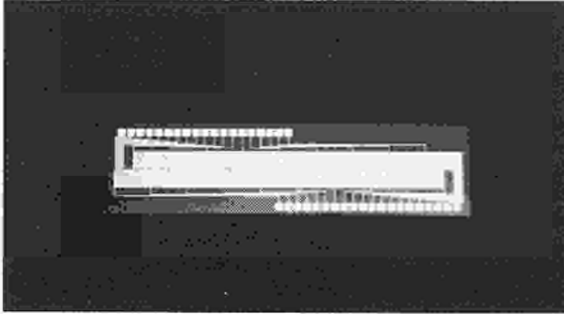


Fig. 5: Realization of a 10 cm/4 pixel scanner with pin photo diodes and separated p⁺- and in⁺-patterning

The feasibility of processing 10 x 10 cm² substrates with 4 scanners with 4 pixel/mm and 10 cm length was shown. There are up to now some problems in the mask alignment, but they can be overcome by a slight modification of the photolithographic masks and the process technique. Some measurement results on these scanners are reported in the following chapter.

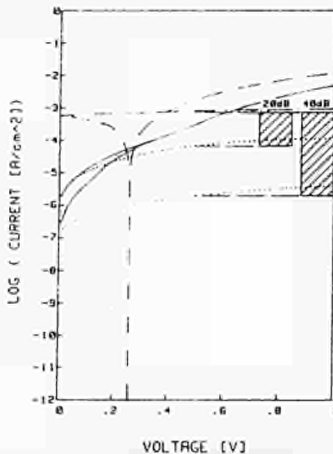


Fig. 6: Dark- and photo current of some pixels of the scanner in fig. 5

photo current: - - - forward, - - - - reverse
dark current: --- forward, reverse

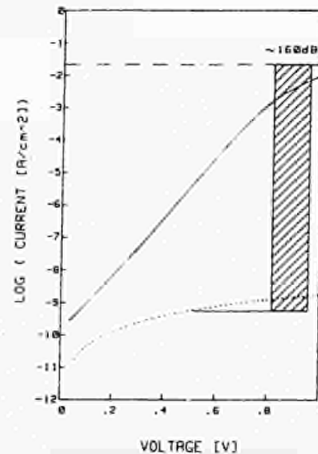


Fig. 7: Dark and photo current of the pin test structure with 0,1 cm² area

2.2 Measurement of Photo- and Dark Current Characteristics

The photo- and dark current as a function of the forward and reverse bias voltage were measured. The measured data are shown in fig. 6.

For measuring the photo current we used a "room" illumination of about 10-20 W/m². At this low light level a photo to dark current ratio of 20 to 40 dB was achieved.

If AM1 illumination level is used the photo to dark current ratio increased to 60 or 80 dB.

The high dark current is caused by the above mentioned misalignments of the photolithographic masks.

For testing the photo- and dark current resulting from the designed pin structure itself, we produced test pin structures with an area of 0,1 cm².

The characteristics of these test structures are shown in fig. 7.

Fig. 8 shows the very simple design of the test structures. As can be seen it seems possible to achieve photo- to dark current ratios of 110 to 160 dB (depending on light level used).

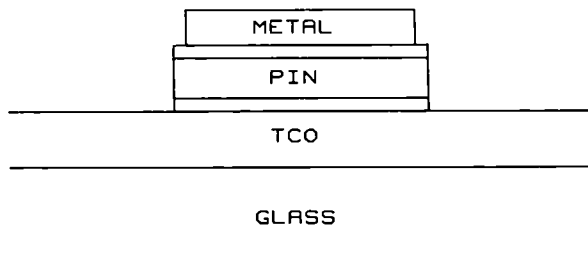


Fig. 8: Cross-sectional drawing of the simple design of the pin-test-structure for measuring the semiconductor dark current limitation

In conclusion we realized a first linescanner with pin-device-structure, 4 pixel/mm and 10 cm length. First photo- and dark current ratio measurement on the scanner are done. We achieved photo to dark current ratios up to 40 dB at a light level of 10 - 20 W/m². We demonstrated photo to dark current ratios up to 160 dB with test structures for the pin device design.

2.3 Read-Out, Interconnection and Packaging of the Contact Imager

As far as the packaging of the sensor is concerned, an approach has been set up which was guided by following objectives:

- o design of a package allowing integration of alpha-Si imager substrates of various length (up to A4), resolution and read-out technique;
- o modularity allowing different combinations such as:
 - imager without lens and illumination
 - imager with lens but without illumination
 - imager with lens and illumination
- o illumination consisting of a double linear LED array
- o as a first prototype, a read-out circuit of the integration pulsed linear type using discrete, dedicated driver chips and dye/wire bonding technology.

A cross-sectional view (fig. 9) shows the basic elements of the sensor assembly:

1. The glass substrate with the alpha-Si sensors and Al-conductor paths provided by MBB
2. The metal package frame
3. The distribution printed circuit board
4. The driver chips, dye bonded on the glass substrate and connected to PCB and Al-sensor conductors by wire bonding
5. Silicon gel coating
6. Package cover

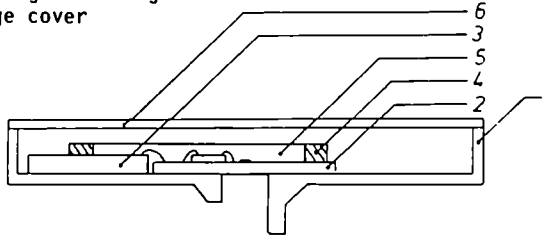


Fig. 9: A cross sectional view of the basic elements of the sensor assembly

An additional package with LED arrays and a linear SELFOC lens can be added as depicted in fig. 10.

Moreover, this package is designed to assure a correctly focussed guiding of the original to be read.

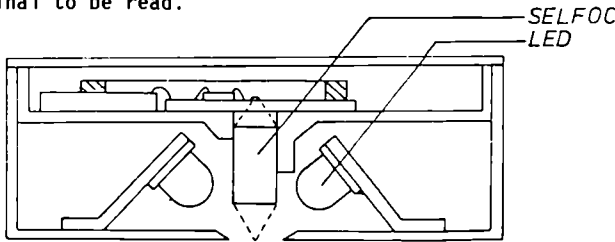


Fig. 10: Additional package with LED arrays and a linear SELFOC lens

For the first prototypes, the alpha-Si imager will be read-out by dedicated driver chips, each consisting of a 64-bit shift register, a 64-bit latch and 64 current switches. As a target value the sensor will be sequentially read out at 1 Mpixel/second yielding a scanning time of about 2 ms/line for a 1.728 pixel array. Above performance figures satisfy quite a large range of possible applications for linear imagers. In the following table global market requirements are summarized with regard to size, speed, resolution and number of detectable grey levels:

Application	Size		Speed (page/min)		Resolution (dpi)		grey levels	
	min	max	min	max	min	max	min	max
OCR, moderate speed	A4	A4	1	2	200	300	2	2
facsimile, gr 3	A4	A4	1	2	200	200	2	2
desktop publishing	A4	A3	2	10	300	400	16	256
electronic copier	A3	A2	5	15	300	400	32	256
OCR, high speed	A4	A4	2	10	200	300	2	2
archival, mod. speed	A4	A3	10	40	200	400	16	64
facsimile, gr. 4	A4	A3	5	10	300	400	2	16
archival, high speed	A4	A3	20	80	200	400	2	16
blackboard copier	A2	> A0	1	5	1	2	2	2

In view of the limited speed performance of alpha-Si sensors only OCR, facsimile, desktop publishing and blackboard copier remain as possible candidate systems for applying this sensing technology. System manufacturers are motivated to switch over from the more conventional techniques for reason of compactness, reliability and cost. Moreover, the emergence of alpha-Si contact imagers opens up new applications like the blackboard copier. The on-going project will be focussed to those developments that yield high quality imagers with flat spectral responses, high dynamic ranges and resolution/speed character.

prototype: 200 dpi / 1.728 elements / 2 ms line time
 future extensions: 300 dpi / 2.540 elements min / 3 ms line time
 400 dpi / 3.456 elements min / 4 ms line time

While the prototype still combines crystalline read-out chips and alpha-Si sensor structure, it is the aim to integrate in a next phase the read-out mechanism in alpha-Si or poly-Si batch which is directly deposited on the glass substrate. This integration of thin film transistors will lead to a much cheaper and less complicated manufacturing processes and hence to lower cost of the sensor.

3. ALTERNATIVE DEPOSITION TECHNIQUES

3.1 Homogeneous CVD: "State of the Art" (by IMEC)

The homogeneous CVD system basically consists of an LPCVD chamber and a cooled substrate holder (fig. 11). The SiH_4 gas is heated up to about 700°C , while the substrate is sitting on a cooled substrate holder at a temperature lower than 350°C . Therefore the surface "heterogeneous" CVD reaction is quenched and film growth is caused by equivalent of plasma glow discharge is obtained, where gas temperature determines the film growth rate and substrate temperature determines the hydrogen content and hence the opto-electronic properties of the films. Because this is a "softer" deposition technique, it is speculated that this method yields more stable films, which is under investigation [1].

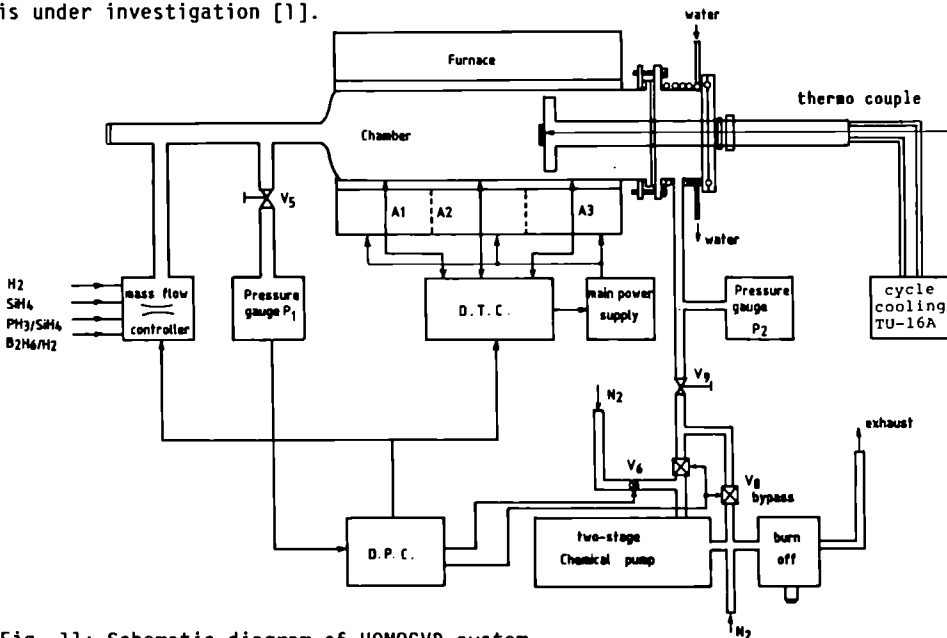


Fig. 11: Schematic diagram of HOMOCVD system

After initial set-up of the system, a lot of modifications had be done and are still underway such as the installation of a cycle cooling system for better control of the substrate temperature, the design and fabrication of a contamination free substrate holder (see further), the design of special clips to support the samples to improve uniformity and the use of gallium for better thermal contact between the substrates and the substrate holder. Assisted by theoretical modelling of mass and heat transport in the reactor, proving that after 1,5 cm of flow after introduction, the silane gas reaches its nominal set temperature, an optimization has been done of the ratio of radius to lenght of the reactor tube, using a movable gas injector and/or shortened tube. Given the diameter of about 14 cm of the reactor tube, the optimal distance between gas injection and substrate holder was found to be around 64 cm, resulting in the highest growth rate. The final gas temperature was installed at 700°C.

The temperature difference between the surface of the glass substrates and the cooled block of the substrate holder has been monitored by thermochrome crayons with nominal temperature calibration. The difference in temperature, when using gallium between substrate and substrate holder was found to be between 20°C and 30°C. Also an investigation of the gas phase and local (at the substrate holder alone) nucleation phenomena has been done in function of process temperature and pressure of SiH_4 (fig. 12). These experiments show that the real limitation is set by the local nucleation phenomenon. Systematical investigation of growth rate, uniformity and optoelectronic parameters of deposited intrinsic layers from silane versus gas temperature, pressure, gas flow rate and substrate temperature have led to the following "state of the art" results:

- growth rate : up to 100 Å/min = 0,6 µm/hr (CVD-like)
up to 400 Å/min = 2,4 µm/hr (PVD-like)
- uniformity over 8 cm 0 : ± 5 %
- dark : $1,8 \cdot 10^{-10} \Omega^{-1} \text{ cm}^{-1}$
- phot(AM1) : $2 \cdot 10^{-5} \Omega^{-1} \text{ cm}^{-1}$
- E_{act} (activ.energy) : 0,78 eV
- E_{g} (optical gap) : 1,75 eV

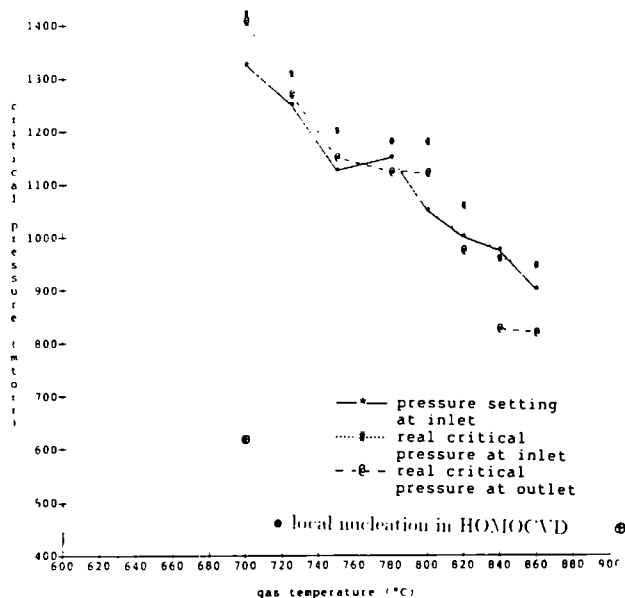


Fig. 12: Critical pressure of gas phase nucleation in LPCVD

Remarkable is the fact that conditions are determined under which "PVD-like" films are obtained and under which "CVD-like" films are obtained (see SEM picture on fig. 13 and 14). PVD-like films show a certain structure in their morphology, while CVD-like films show a complete smooth featureless morphology resulting in better uniformity, but somewhat lower deposition rate. The transition between the two morphologies is determined by gas temperature and process pressure. The conditions are to obtain good "CVD-like" films.



Fig. 13: Morphology of PVD-like amorphous silicon films in HOMOCVD

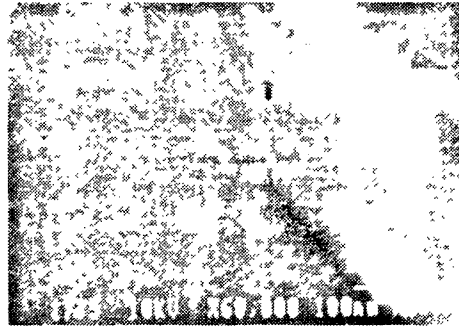


Fig. 14: Morphology of CVD-like amorphous silicon films in HOMOCVD

A problem which has to be solved is unwanted contamination from the stainless steel substrate holder. This was determined by SIMS measurements. Cr was found in the films up to a concentration of 10^{17} cm^{-3} . A new quartz substrate holder is currently being made to overcome this problem.

Near future research will continue on doping of the a-Si:H-layers and on investigation of the photostability. Also the homogeneous CVD material will be compared with a-Si deposited by plasma glow discharge for their use in contact imagers. A set of masks has been designed for a limited size sensor array of 128 sensor elements (8 pixels/mm).

3.2 Comparison of the Processes Induced by Mercury Lamp and ArF Eximer Laser Photoassisted CVD of a-Si:H Film (by CNRS)

The a-Si:H films were prepared by photodissociating SiH_4 molecules sealed in a reactor chamber using both a low pressure mercury lamp and an ArF eximer laser. Important differences have been observed especially during the first stages of the deposition induced by these two different types of excitation sources. With the mercury lamp the process may be characterized by a long period of incubation for initiating film growth. In this case the deposition rate depends linearly on the light intensity:

Light intens. at 254 nm Ref. (mW cm^{-2})	Dep.rate* (\AA min^{-1})	Dep.per light int. ($\text{\AA cm}^2\text{min}^{-1}\text{mW}^{-1}$)
20	30	1.5 [3]
10	12	1.2 [2]
1.5	3	1.5 This work

*measured at about 1 Torr SiH_4 pressure

By contrast, with the ArF laser the deposition rate appears to be strongly non-linear with the laser power (Fig. 15). We measure very high deposition rates ($\sim 3.000 \text{ \AA}/\text{min}$) for the first 200 Å, which decrease for thick deposited layers ($> 1.000 \text{ \AA}$). The existence of these two different regions in the growth rates has not yet found a satisfactory explanation.

Film deposition rates also strongly depend on SiH_4 pressure (fig. 16). With mercury lamp the initial deposition rate begins to increase above a pressure of 0.1 torr, reaching a maximum value around 20 torr, before dropping off rapidly at SiH_4 pressures higher than 50 torrs. A similar behaviour is observed with the ArF laser. In this case, the maximum deposition rate is reached at about 100 torr of SiH_4 before dropping off at higher pressure by the formation of large amounts of powdery silicon.

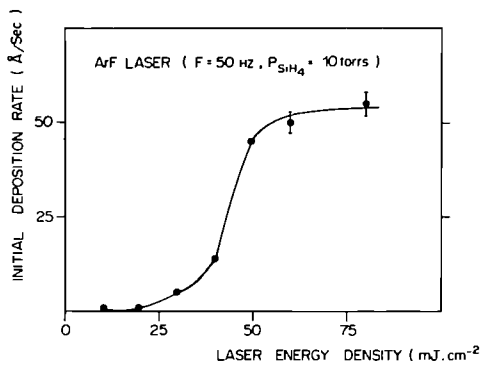


Fig. 15: Initial deposition rate as a function of ArF laser energy density

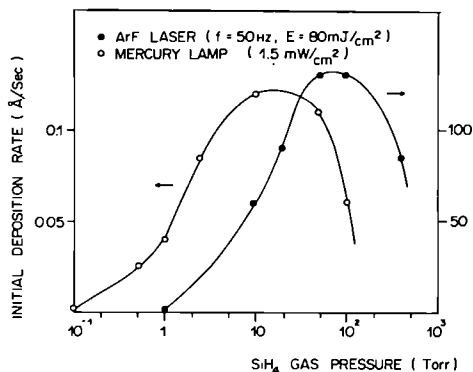
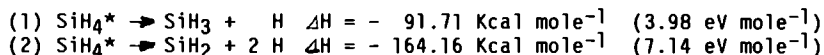
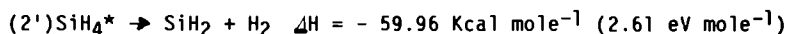


Fig. 16: Initial deposition rates as a function of SiH_4 pressure for ArF and Hg lamp excitation light

The experimental results show that UV photoexcitation of SiH_4 molecules can be achieved through two different pathways which depend on the wavelength and intensity of the light source. With a low pressure mercury lamp, Hg atoms mixed with the SiH_4 are excited by the 254 nm resonance line and transfer their energy to SiH_4 molecules by collision. With the ArF excimer laser, photons of 6.4 eV (193 nm) are not able to directly dissociate SiH_4 by single photon excitation, because the significant absorption starts only above 7.8 eV (160 nm). However the decomposition of SiH_4 by a two photon excitation of molecular electronic states might occur above a laser threshold energy of about $30 \text{ mJ}/\text{cm}^2$ as can be deduced from experimental results of fig. 15, which show the non-linear behaviour of the process. In either form of excitation, the nature of the decomposition of photoexcited SiH_4 molecules into reactive species and intermediate radicals remains a subject of conjecture. Two main primary reactions have been proposed [4] [5].



A third reaction:



sometimes mentioned, appears to be energetically more favorable than reaction (2). In fact, it may not be considered as a primary step since it results from reaction (2) followed by the exothermic recombination of H radicals releasing $104.2 \text{ Kcal mole}^{-1}$.

of noise considerations. Indeed, the long interconnections, needed to connect the sensors with the driver IC's, cause large coupling capacitances, which will have a detrimental effect on the sensitivity of the image sensor (fig. 17) [6]. This parasitic capacitance can be largely reduced by integrating the driver thin film transistors together with the a-Si:H-sensors. In this way the interconnection length can be reduced by a factor of 100, making thereby this capacitance negligible to the sensor capacitance.

However the question arises then in which material the thin film transistor should be made; a-Si:H or polysilicon. Some straightforward calculations, based on simple MOS-transistor models, and taking into account the nowadays requirements for speed (25s for reading an A4-page) and resolution (8 pixels/mm) learn that the mobility for the driver thin film transistors has to be in the order of $1 \text{ cm}^2/\text{Vs}$ for a ratio W/L equal to 5. This would allow the use of amorphous silicon. By integrating also the shift register and some simple logic circuitry on the same substrate, the number of hybrid connections can be reduced substantially, making this solution cheaper. Taking this option, the thin film transistors need to be made in poly silicon due to the higher mobility requirements. At the moment, four ways for obtaining polysilicon on a quartz or glass substrate are under investigation:

- a) Direct deposition of polysilicon by LPCVD on quartz (IMEC)
- b) Thermal recrystallization of LPCVD amorphous silicon on quartz (IMEC)
- c) CW-laser recrystallization of a silicon layer on glass (IMEC)
- d) Pulsed laser recrystallization of amorphous silicon on glass (CNRS)

The results of each technique will be dealt with in more detail in the following sections.

4.1 Direct Deposition of Polysilicon by LPCVD on Quartz

This material was deposited at 610°C . We characterized it by dark- and photo conductivity measurements, optical transmission, TEM and SEM. This way of fabrication resulted in a non-uniform material - the polysilicon consists of a lot of small grains $< 50 \text{ nm}$ and some big crystallites of $0.5 \mu\text{m}$ - with a rough surface. Therefore this material is less useful in TFT's because this would lead to a rough gate insulator having a low breakdown voltage.

4.2 Thermal Recrystallization of LPCVD Amorphous Silicon on Quartz

In this method first an amorphous silicon layer was deposited by LPCVD at 550°C followed by a thermal recrystallization in a conventional furnace in N_2 at 975°C . We did the same kind of characterization as for the direct deposition of polysilicon (4.1). By this technique we obtained a reproducible, uniform, rather large grain polysilicon (grain size about $0.5 \mu\text{m}$) (fig. 18). The surface was smooth and the layer did not contain cracks even when directly, deposited on quartz. Hence bufferlayers between the polysilicon and the quartz substrate are not necessary, which substantially facilitates this process.

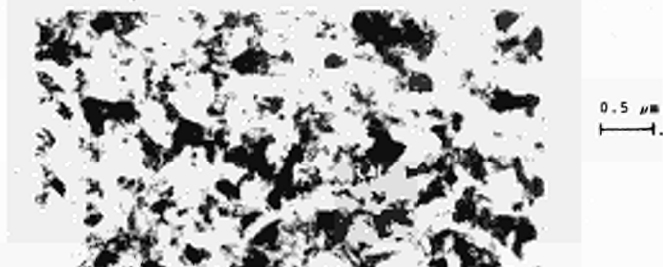


Fig. 18: Thermal recrystallized amorphous silicon

We have started the fabrication of polysilicon TFT's using this method. One sees a cross section in fig. 19. Because this process requires high temperature steps, the substrate must be quartz. In the future this high temperature process will be optimized. On the other hand we will also try to lower the temperature of the different steps so that much cheaper glass substrates can be used.

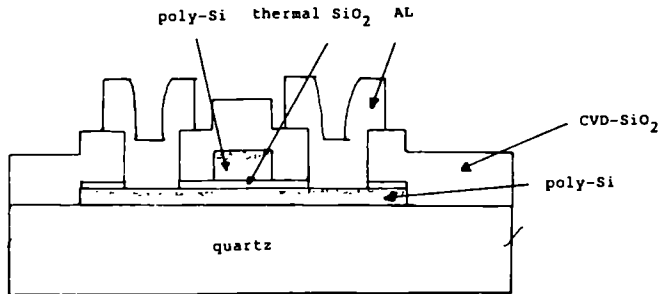


Fig. 19: Cross section of TFT

4.3 CW-Laser Recrystallization of a-Si:H or Polysilicon on Glass

In this third recrystallization technique a CW Argon-ion laser, working at 500 nm, was used to recrystallize silicon thin films. By this method large grain poly silicon (5 μm) can potentially be obtained. For the moment we have however focussed on the elimination of large microscopic defects as cracks, balling up of the molten layer and mechanical damage to the glass. The material analysis was performed by SEM and Nomarski interference microscopy. Measurement of the electrical conductivity and activation energy was done to get a first idea about contamination, coming from the glass. Our experiments on the first kind of structures (fig. 20) indicate that recrystallization suffers from problems related to the inherent instability of the amorphous silicon layer.

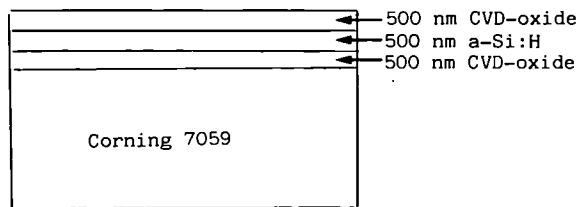


Fig. 20: First layer structure for studying laser recrystallization

Although the transmission of the films is clearly enhanced after recrystallization - amorphous silicon has a much larger absorbance ($10^5/\text{cm}$) than crystalline silicon ($10^4/\text{cm}$) - the material is found to consist of very fine grains ($< 50 \text{ nm}$). Besides this effect a real melt of the whole layer at higher laser powers could not be obtained because of heavy damage to the glass substrate (the glass becomes actually diffuse). This damage can however be minimized by using a plasmanitride layer of typically 100 nm between the glass and the CVD-oxide layer, thereby also reducing the contamination coming from the glass, which was confirmed by activation energy measurements.

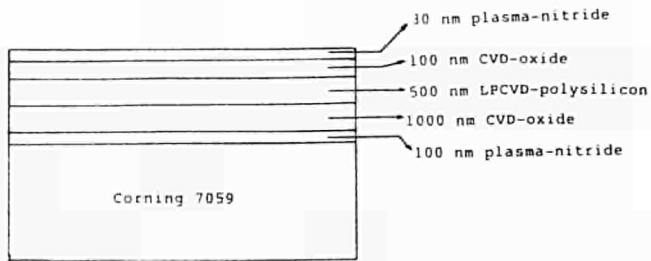


Fig. 21: Most promising layer structure for laser recrystallization

The most promising results were obtained when recrystallizing polysilicon, deposited at 610°C, and using the multi layer scheme of fig. 21. The enlarged grain size can be seen in fig. 22 and 23. In this structure the problem of cracking was not completely solved, but a detailed study reveals that the cracks are typically spaced a few 100 μm (fig. 24). Therefore the recrystallization of small islands will be pursued.

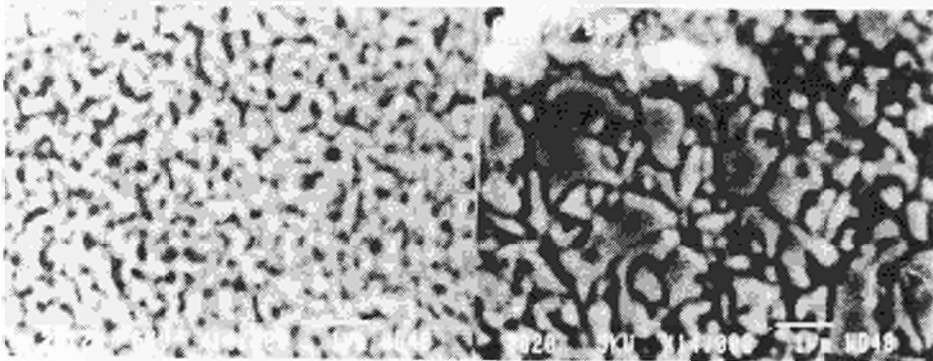


Fig. 22: Unrecrystallized polysilicon

Fig. 23: Laser recrystallized polysilicon



Fig. 24: Typical crack pattern after recrystallization

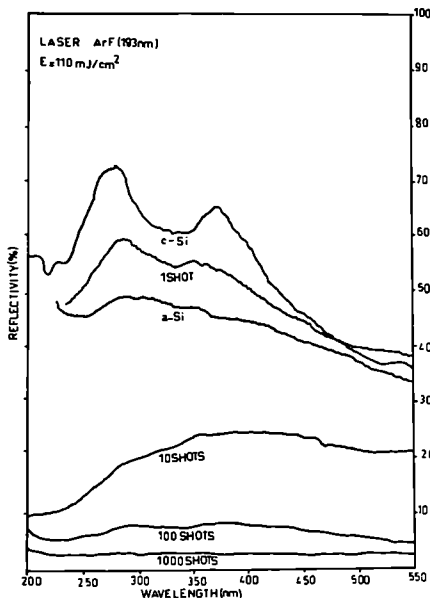


Fig. 25: Reflectivity spectra, showing the recrystallization of the amorphous layer

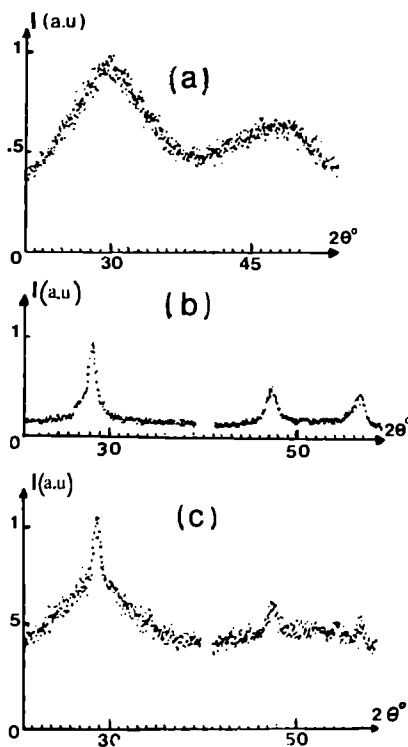


Fig. 26: Scattering profiles for the 3 different samples studied with $R = 0,9^\circ$

The X-ray scattering experiments, performed at a grazing angle appears especially well adapted to the study of the very near surface regions. As shown in fig. 26, where the scattering profiles are recorded as a function of the diffraction angle 2θ for three different samples, the amorphous layer (fig. 26 a) appears to be completely recrystallized after a laser shot of 130 mJ/cm^2 (fig. 26 b) (3 diffraction peaks corresponding to the $\langle 311 \rangle$ of crystalline Si). For a laser energy density of 110 mJ/cm^2 , the diffraction profile (fig. 26 c) which shows both the amorphous and crystalline signal, indicates that the layer is only partially recrystallized. The crystalline part is probably at the surface because tuning the grazing incident angle towards smaller angles increases the crystalline

response. TEM results confirm the results of the X-ray study, as shown in the micrographs (fig. 27 a, b). The layer is clearly completely recrystallized with grains elongated in a direction perpendicular to the silicon surface.

4.4 Pulsed Eximer Laser Recrystallization of Amorphous Silicon Thin Films Deposited on Glass [7]

In this work the influence of a pulsed excimer laser, working at 193 nm (ArF) on the structure of amorphous silicon thin films, deposited on glass was investigated. The structure was essentially the same as in fig. 20. Different characterization techniques such as visible and UV reflectivity, X-ray scattering and transmission electron microscopy have been compared to obtain a better understanding of the recrystallization process. As shown on the reflectivity spectra of fig. 25, the recrystallization of the amorphous layer may be well characterized by the presence of the two peaks E1 and E2 respectively located at 3.4 and 4.5 eV.

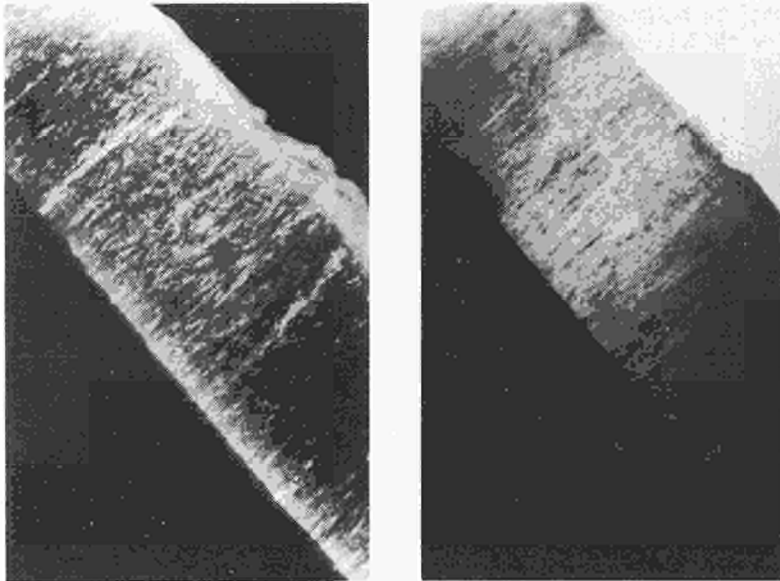


Fig. 27: TEM pictures of the recrystallized sample (a), (b)

5. CONCLUSIONS

As already mentioned in the introduction this project contains a development part and a fundamental research part. The outcome of the two years feasibility phase will be on one hand a first prototype of a packaged linear sensor array (A4 size) with 8 pixel/mm and a clear view on possible market applications. On the other hand this project phase will allow to judge the value of alternative deposition techniques for the amorphous silicon sensor elements in comparison with the plasma glow discharge technique. Furthermore polysilicon TFI's will be realized on glass and/or quartz for later integration as drivers/shift registers together with the amorphous silicon sensor elements.

This will allow to converge the effects in a later phase awards the realization of an endproduct, possibly with complete integration of the logic circuitry on the same substrate, avoiding any hybrid connection between sensor elements and driving logic.

REFERENCES

- [1] B.A. Scott, Homogeneous Chemical Vapor Deposition, Semiconductors and Semimetals, Vol. 21, part A, Chapter 7, ed. by J. Pankove, Academic Press Inc. (1984), p. 123.
- [2] J. Perrin, T. Broekhuizen, R. Benfehrat, In Laser Processing and Diagnostics II, edited par D. Bauerle, K.L. Kompa, L. Laude, Les Editions de Physique (1986) p. 129.
- [3] N. Mutsukura, Y. Machi, Appl. Phys. B41, 103 (1986).
- [4] H. Niki, G.J. Mains, J. Phys. Chem. 68, 304 (1964).
- [5] G.G.A. Perkins, E.R. Austin, F.W. Lampe, J. of Am. Chem. Soc. 28, 1109 (1979).
- [6] Kouichi Seki, Tatsuji Matsuura,
Analysis and design of a large scale image sensor using amorphous silicon,
IEEE Transactions on components, hybrids and manufacturing technology
3/9/86
- [7] A. Naudon, J.C. Desoyer, E. Fogarassy, J. Maillou, E.L. Mathe,
T. Slimani
Symposium on "Photon, Beam and Plasma enhanced Processing", E-MRS,
Strasbourg, 1987

An office assistant prototype

using a knowledge-based office model

on a personal workstation

Martin Ader, Michel Tueni

Bull MTS, Direction des Etudes Avancees

1 rue Ampere, B.P. 92, 91301 Massy CEDEX FRANCE

ABSTRACT

The Intelligent Workstation ESPRIT project B 82 aims at the development of a complete system providing assistance to office workers. The components of the project are: a table-top workstation with a graphic oriented UNIX operating system, natural language processing tools, a user interface management system and a knowledge representation system. The application assists the user in defining its own office context in terms of the office model at the level of detail he wants. The office process assistant uses this definition to monitor office processes that can be recognized by executing office operations for the user whenever possible or by requesting user actions when needed. The present state of the implementation underlines the importance of explicit representation of all aspects of the office environment including organization and procedures as well as the importance of the user interface for easy description of its context by the user. The success of such an approach would bring a major step in office productivity by providing assistance at the level of office processes made possible by the explicit representation of the organization and procedures in the office information system.

1 Introduction

The intelligent workstation ESPRIT project (B 82) aims at the development of a complete system providing assistance to office workers. After a one year pilot phase, the project started in March 1985 and will end in February 1989. Associated partners are: Bull in France (main contractor), Katholieke Universiteit of Nijmegen in Netherlands (KUN), OCE in Netherlands, Research Center of Crete in Greece (CRC), and Vrije Universiteit of Brussel in Belgium (VUB). The workstation hardware (Bull) provides a high quality table top office engine with graphic and voice support. The operating system environment (Bull) is UNIX based with efficient graphic and audio support and large LISP processes support. Communications through local area network and phone interface is provided.

A User Interface Management System (CRC) will enable easy development of user interfaces for the targeted applications. UIMS will provide application interface independence and dynamic interface modification for rapid prototyping. Natural language facilities will be provided (KUN and OCE) in Dutch and English. An author system with linguistic knowledge has been implemented and will be followed by a dialog system used for querying the office

application status in natural language. The Knowledge Representation System (VUB) has been adapted to the workstation environment and will be further extended with new formalisms needed by office applications.

Office applications (Bull) will enable explicit representation of office objects, procedures and organization. Using this representation, applications will provide guidance on what to do in each situation and assistance by executing some of the proposed actions and monitoring user activities. Applications will be built upon KRS and use the UIMS and Dialog System for user interface developments.

We present here the first prototype of the targetted application of the workstation. We will first present the applications goals of the project and of the prototype. Then , after an overview of the architecture chosen for implementing it, we will successively present the office model, the activity manager, the top level manager and the user interface. A presentation of future work will close this overview.

2 Application goals

Present office applications correspond to basic terminal tasks of the office work: text editing, document filing and retrieval and document mailing. The primary goal of the project is to demonstrate the feasibility of a knowledge based office process assistant application taking into account not only these basic tasks but the whole process that required them. This implies a model of office processes and the implementation of formalisms to represent them. Using these formalisms, three levels of applications will be built: office process analysis, office process advisor and office process assistant. These applications should:

- enable office activities representation including documents, organization and procedure;
- use that representation in order to provide advice on actions requested by each situation;
- whenever possible, assist the user by executing for him required actions that can be automated and monitoring subsequent actions until goal achievement.

The bibliographic review of office models and the fields studies of clerical and managerial processes have led to a first list of concepts to be represented: office facilities, organization, information, procedures. The aim of the Intelligent Office System layer, built above KRS, is to provide the means of describing these concepts and their mutual relationships. This IOS will act as a meta-model of office world and offer control structures tailored to this environment. The kernel part of the IOS is an Activity Manager System (AMS) which allows the declarative representation of an activity network in terms of states, activities and actions. An evaluation mechanism is provided on this representation, taking benefit of the concepts introduced as: enable state, caused state, activity elaboration and enable links.

The office process analysis application will provide office users with the capability to describe their processes in terms of the IOS model without the required IOS/KRS/LISP expertise. The office process advisor application will use an office description to provide user advice on what to do in order to achieve office goals in each situation. The office process assistance application will provide to the user execution assistance and monitoring of process instances.

The office assistant prototype presented here-under corresponds to the office process assistance application of the project. It uses a first version of the IOS model and of the AMS built upon KRS and a user interface based on tools developed by Bull.

3 Application architecture

The office assistant prototype application architecture is built on an office knowledge base which contains a declarative description of the context of the office covering both static and dynamic aspects. This office knowledge base contains an office meta-model gathering a general model of any office environment including; organization, office facilities, informations and procedures. This meta-model is used to build an office model of the specific office of the user of the workstation. This office model contains a description of the organization environment of the user, of the procedures that he can use and of the facilities available to him. The office context gathers all the informations created or received by the workstation as well as the state of on-going processes.

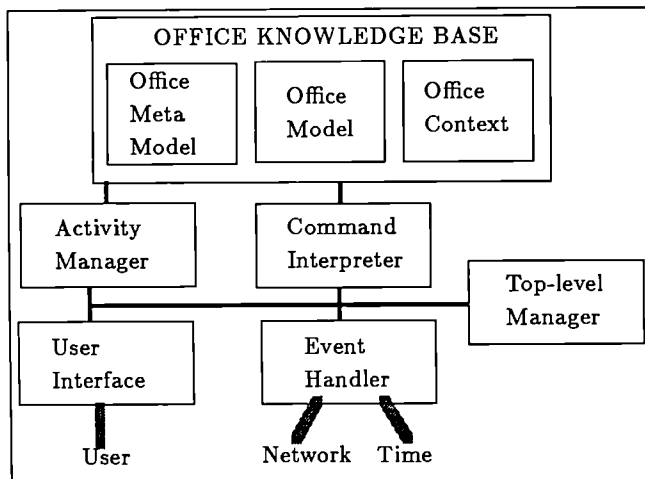


Figure 1: Application architecture

Given this knowledge base, the application offers to the user a set of commands classified into three categories:

- (i) Operators corresponding to classical office operations like: create a document, file a document, send and receive letters and messages.
- (ii) Office operators corresponding to higher level operations in an office like: request a service, obtain approval, request comments, revise a document, control a document.
- (iii) Procedures corresponding to operations defined in the company in a standard way like: buying equipments, traveling, hiring a new employee.

Each of these commands are implemented in the form of a network of activities whose description is contained in the knowledge base. To each of these activities corresponds an action that implements it. These actions use the knowledge base and interpret the semantic of the informations as described in the base. They modify the knowledge base in order to reflect the result of the action.

When the user executes a command through the top level manager, it creates a process whose execution is monitored by the activity manager. The activity manager monitors the process execution by interpreting the activity network description corresponding to the requested command in the context set by the commands arguments. Actions corresponding to activities of the network that must be executed given the context are executed and can update the

context. The context of the process is maintained in the context area until the process is completed.

An event handler monitors two categories of external events: time and received documents. When a document is received through electronic mail, the event handler analyzes it. If it is an answer to a previous request the suspended process corresponding to that request is resumed. If it is a document recognized as a request to initiate a procedure, a new process is initiated with that procedure and the received document as argument. Time events are processed in a similar way, the process waiting for the time event is resumed.

All user interactions use the same user interface system (UIS). Built upon the X-window server, the UIS offers multiwindowing capabilities, menu command entry, graphic structure edition and data display and entry through forms.

4 Office model

4.1 Overview

The role assigned to the Intelligent Office System is to provide the office model allowing the representation of each specific instance of an office environment. General requirements for that representation are the followings:

- enable progressive definition of each office environment;
- corresponds to concepts which are those currently recognized by office workers;
- definition should be explicit for easy modification by specialists;
- full integration into the development environment for easy extension by specialists.

The present state of the implemented model includes: organization, information, facilities, procedures and time.

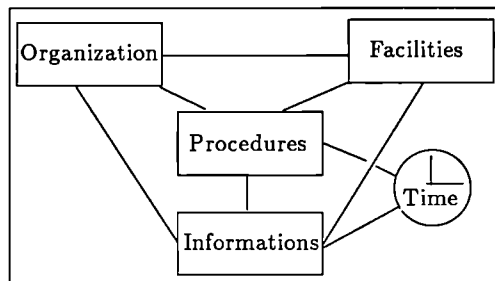


Figure 2: IOS model overview

Organization concepts represented are: organizations (as Bull, Vub, OCE), organization-units (as AI Lab, DEA, Marketing), persons, organization roles (as manager, president, project leader), project structures (ESPRIT 1, ESPRIT 2, EUREKA, IWS). These concepts structures are organized in a way allowing the definition of office objects as: the IWS project leader (the person holding the role project leader of the project IWS); the employee manager (the person holding the role manager of the organization unit to which the employee belongs to).

Facilities represented are: top of desk, desk, mailboxes and personal files. Informations modification are prohibited outside the top of desk area, informations residing in the desk area are considered in a some thing to do about state, informations residing on permanent storage are considered in a not to be updated state. Information objects include informations supports found in an office environment: documents, letters, forms .

Procedure representation enables procedure description by declaring: procedure roles, activities, documents, and their mutual relationships. Predefined procedure roles are offered (approval and control). An authority definition mechanism is provided in order to specify approval levels for each approval procedure role. Predefined activities are provided. They implement repetitive simple activities networks involving office operators like: request document approval, request document control, monitor a date as a deadline. Activity representation enables the specification of complex network of activities at different levels of abstraction.

Time representation includes: time point, time interval, time units (day, month, year), and operations on these concepts.

4.2 Organization

Organization model describes persons, their roles and the way they share their responsibilities in structured organization-units. The top level concept is the organization concept with two specializations: party and project. Party refers to persons or organizations that constitute an entity, project refers to a combined structure involving several parties.

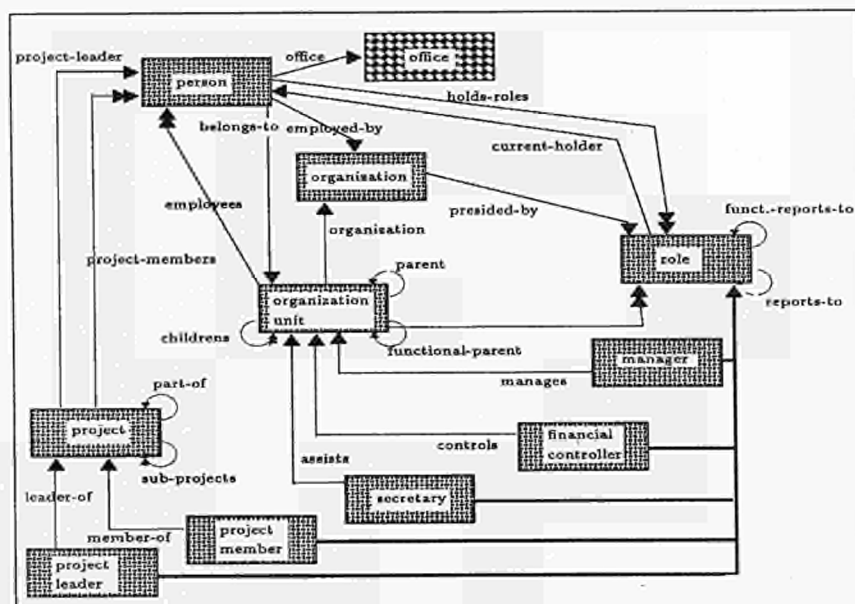


Figure 3 : Organization Model

The party concept is specialized by organization, organization-unit, role and person concepts. The organization concept describes a legal entity as a company or a university or an administration. The organization-unit concept is the basic building block used in describing hierarchical organization structure of an organization. The role concept refers to some recognized roles within the organization, it is the basis for links with the dynamic part of the model. The person concept refers to an individual who is part of an organization.

The person concept is further specialized with two concepts: organization employee and organization-unit-employee. These concepts are introduced for facilitating description. Organization employee concept introduces standard definition of manager, financial-controller

and secretary taking into account the organization-unit concept structure. Specializations of role concepts are proposed: manager, financial-controller, secretary, project-leader and project-member. They are related to organization concepts and correspond to roles frequently used in procedure descriptions.

4.3 Facilities

The main material handled in an office is information in the form of collections (documents, forms, letters). These informations come either from outside, or are created by the office worker himself for its own usage or in order to send them to others. These collections are stored in office facilities.

In the office, collections are stored in repositories. We define two classes of repositories: the permanent repositories where information is kept for a long period and must not be updated, and the temporary repositories where informations are stored for later usage. All communications go through two repositories: the in-tray repository and the out-tray repository. In and out trays are viewed as repositories where informations are stored by the communication system for the in-tray, and by the user for the out-tray. Retrieval from the in-tray is done at user request and from the out-tray by the communication system. Informations in the trays cannot be updated.

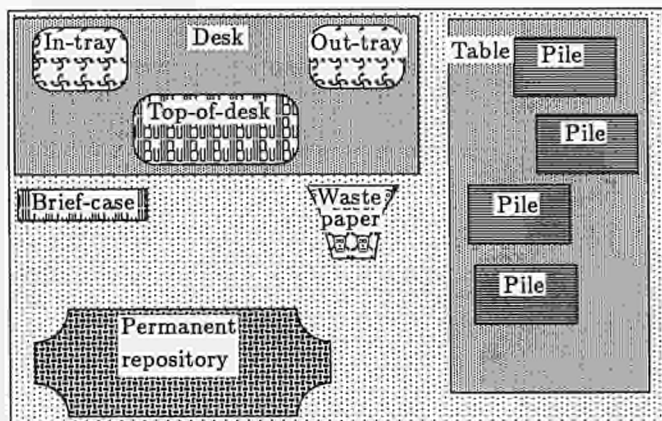


Figure 4 : Facilities Model

When the office worker is working on collections of informations, he puts them in an area of its desk that we call the top-of-desk area. It is where he puts communications extracted from the in-tray in order to analyze them and decide of their usage. It is from the top-of-desk area that he transfers them to the out-tray. The top-of-desk area is the only repository where collections can be created or modified or presented by the user. Collection modification is forbidden for collections which are not created by the user itself.

The desk of the user is a temporary repository where he can store collections in more or less ordered piles for later usage or in order to visually remind work-to-do-later. Some users need an extension to their desk area in the form of a table where they can store less urgent or frequently used collections. These collections are always considered in a something-to-do-about state (to read, to write a report about, to file later, in case some one request it, etc.).

4.4 Time

The main concepts introduced for time representation are:

- time, a duration expressed by a measure with a precision (in seconds);
- time-point, representing a point in time relative to a reference date;
- current-time-point, representing the current time;
- time-interval, representing a pair of time-points;
- periodic-time-point, a time point occurring at regular intervals.

Time is defined as being a basic time expressed by a measure given in a time-unit, with a precision (a basic time). Lower and Upper of time are the basic time representations of limit values of a possible time inside the given precision. Addition and subtraction operations are defined on time. They represent a time whose referent is the result of the operation on referents of operands, with a measure expressed in the lowest time-unit of the time-units of the operands and with a precision taken as the worst precision of the operands (Addition of years to seconds will give a time expressed in seconds with a year as precision).

A time-point concept is a time (with unit and precision) measuring a distance since the reference date of the operating system. Future, present and past are predicates to situate the time-point relatively to the current-time (present returns () if the current date is not in the upper/lower interval of the time representing the time-point). Current-time is a time-point whose referent is obtained from the system each time it is accessed. It is expressed in unit second and precision second.

Time-interval is a time-concept representing two time-points: start and end. Upper of a time interval is a basic-time equal to the upper basic-time of its end time-point. Lower of a time-interval is a basic-time equal to the lower basic-time of its start time-point. Duration is a time measuring the duration of the time-interval in units of the lower unit of start and end time-points and with their worst precision.

A periodic-time-point is a time-concept representing a time-point occurring at regular intervals. Period is the distance between two occurrences. Initial-time-point is a fixed-time representing the time-point of the first occurrence. Next gives the presently fixed next periodic-time-point. Set-next is an action subject setting next to the next time-point (addition of one period). Set-future is an action subject setting next to the next time-point immediately after the current-date.

4.5 Informations

At the top level of the model we define office informations as being collections concepts. We refers to collection as a set of elementary informations gathered together for convenient usage in the office environment.

A document is a collection with its final presentation. It is the form that the user has given to a collection for further communication. As such, it cannot be modified even by its author. A document has an existence in itself, its usage cannot be completely predicted in the system. A document is described from several points of views:

- its identification, as a collection having its own existence;
- its source, identifying how it came in the office;
- its location, locating it in the office;
- its presentation, specifying how it has to be presented;
- its semantic description, describing the role of informations it contains.

A communication is a collection of documents intended to be communicated from a party to a party. The collection is enclosed in an envelope. In order to enforce communications consistency, we constrains the collections part of a communication to be documents (they cannot be modified by anyone including the creator of the collections).

Specializations of document are: report, communication-document, letter, business-letter, internal-letter and message. A communication document is a document explicitly created for communicating informations, it contains the list of persons to which it has been communicated. A message (or memo) is an informal communication document intended to be communicated to a party. It is not logged as a letter is, it contains identification of source and destination as well as attached documents identification if needed. A letter is a communication document specifically created to communicate informations to identified parties. The content of a letter identifies as informations the from and to parties of the future communication as well as the documents part of its collection.

Several collection types are defined to represent specific collections reflecting operations currently used in an office: agreement, approval and evaluation. These collections are referred to as semantic descriptors. An agreement describes informations of a document reflecting a common position between several parties. It has the role of reference in case of conflict between parties, its form is not defined as it does not corresponds to some procedure enforcing the agreement execution. An approval refers to an information, part of a document. It reflects the agreement of a role responsible in the organization for the matter involved by the document's content. An evaluation is a collection, referring to an other document, expliciting the findings of the author about its content.

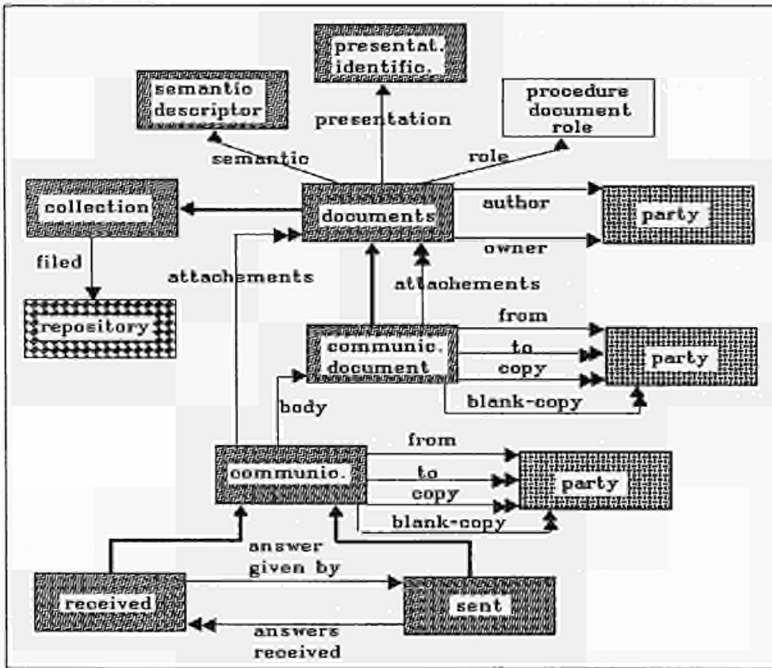


Figure 5 : Information Model

5 Activity manager

The first version of the Activity Manager System is basically an expert system "shell" designed to support applications of a similar nature, in our case Office procedure description and execution. The AMS includes the following elements:

- (i) A representation of office processes in terms of activities and states which captures the sequence of activities that must be executed in order to achieve a goal.
- (ii) Monitors and events mechanisms used for exception processing.
- (iii) An evaluation module including control structures and inference tools in order to trigger the execution of office processes according to their description.

A detailed description of the AMS can be found in [28].

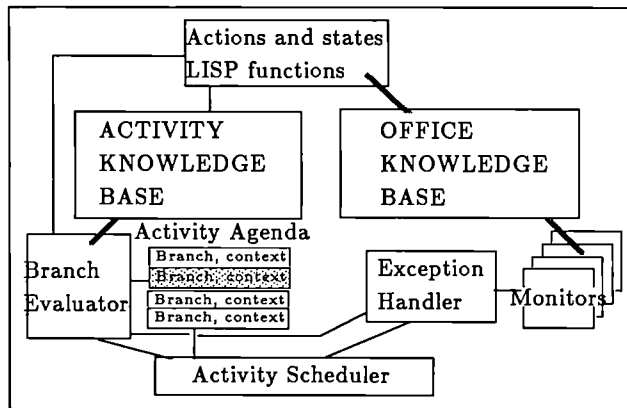


Figure 6: Activity manager architecture

5.1 Basic concepts

The major concept used is the Activity concept. It represents an elementary piece of action in the form of a concept gathering:

- an action (a lisp form executing the action or a refinement of the activity)
- a start state whose evaluation must return true before executing the action
- a caused state whose evaluation must return true after activity execution,
- a set of variables representing the context used for action execution and state evaluation.

An activity instance is invoked with arguments, its action is executed in the environment defined by the arguments if its start state evaluates to true and its caused state to false. As an example, a file activity could be defined with variables `document-to-file` and `where-to-file-it`, with start state `document-must-exist` and caused state `document-filed-in-location` the action being a lisp form filing the document in the specified location.

States are themselves concepts. A state is a concept with a subject evaluation (a lisp form returning true or false) and variables transmitted to the evaluation form. The evaluation subject can be replaced by a list of substates and a logic operator (OR or AND) combining them, the result of its evaluation is then the evaluation of its substates gathered with the logic operator.

An action of an activity can be replaced by a subnetwork of activities. A subnetwork is a structure defining a network of activities that must be executed as an action of the activity it represents. The network is described by a series of activities instances linked together by enable links. An activity instance of a network is enabled-by a list of activities instances with

an OR condition (one of the activities must be executed) or an AND condition (all activities must be executed). This activity, when executed, enables a list of activity with an OR condition (the first eligible activity of the list) or with an AND condition (parallel execution of all the activities of the list). The network includes a specification of the transmission of arguments into the network.

As an example of a network, we can specify the file activity as a network of activities as follows (in an over-simplified syntax):

- (1) Start-file (document location) then OR (2) (3) (4)
- (2) Store (document location) then (5)
- (3) Find (location) then (1)
- (4) Find (document) then (1)
- (5) End-file (document location)

If document and location are unknown, the Store activity cannot execute because its prestate is not satisfied, Find (location) is enabled and its action returns a location to the network, then Find (document) is enabled and its action returns a document and Store is then executed.

Any activity already defined can be used in a network definition, any state can be used in a state definition (recursivity). This gives an abstraction mechanism as well as an "activity library" capability. Above that structure a level of definition has been added called MOPA (Memory Organization Packet for Activity) and inspired from the work of Schank on MOPS. This level enables the specification of skeleton networks that are specialized when they are used by specifying which activity must be used for each node of the network. A MOPA represents an abstract knowledge that can be reused in different situations.

5.2 Monitors and exception processing

With the activity concept as described above, all actions must be completely specified including exceptions otherwise the activity execution terminates abnormally in unexpected situations. In order to give to the activity manager a user driven exception capability the KRS monitor concept is used together with an event handler.

The monitor concept is a KRS concept associated to the built-in consistency maintenance mechanism of krs. A monitor can be associated to any access of a concept (or attribute in more classical terminology). To this monitor is attached a condition and an action. When the value of the access of the concept to which the monitor is attached changes, the condition is tested and if found true, the action is executed. As part of the action the monitor can be deactivated or reactivated.

When the activity manager evaluates a state, if the state is not true, and if this stops the evaluation process, the activity manager tries to analyze the state in order to find if information used to evaluate the state can be monitored for possible future evolution. If it is the case, the activity manager installs a monitor on the corresponding information, suspends the evaluation of the process and informs the user of this situation. The user can then do whatever he wants in order to change the information that is monitored. As soon as the information has been changed, the activity manager will be reactivated by the monitor and the failing state will be re-evaluated.

The states that can be monitored by the activity manager correspond to the following categories of events:

- waiting for a document (to be received later)
- unknown information (to be provided by the user)
- future time-point.

Unknown information monitor is activated when the information is assigned a value. This can

be done by the user by revising the concept which contains the unknown information. The top-level managers maintains the list of such events in a user-things-to-do-list with the identification of the information missing (concept, access) and the process waiting for that information. Waiting-document monitor is activated when the document is received by the workstation. For that purpose, a received-document handler is provided. It analyzes each received document in order to filter those that correspond to events waited by processes. Similarly a time agenda handler detects future time-points that become present and activates the corresponding time monitors.

5.3 Activity evaluation

When an activity is invoked with arguments, the activity manager evaluation process starts. It first evaluates the start state of the activity, if it is true, then the action of the activity is executed (or his network evaluated) only if the caused state of the activity is not verified (if the caused state is true then the action is not needed and the activity is considered as executed)

If the activity is represented by an activity network, the activity manager evaluates its network in the following way:

- it executes the first activity of the network;
- this activity, when executed, enables its enable activity list;
 - if it is by an OR, the first activity that can be executed is executed;
 - if it is by an AND, all the activities of the list are enabled for execution (several execution branches are created);
- when all the branches generated by that process are executed, the activity is considered as executed;
- an activity of an activity network enabled by the activity manager is candidate for execution as soon as one of its enabled-by activities are executed in the case of an OR and as soon as all of its enabled-by activities are executed in the case of an AND.

Thus the execution of an activity invocation generates possibly the execution of several branches of its network (each activity of that network can be itself represented by a network of activities). The context of the execution of the process corresponding to an activity invocation is thus a list of enabled branches with for each branch a specific context made of the variables and arguments bindings for current activity and states for the branch.

At the level of the evaluation of a branch, when the state evaluation fails, the exception handler of the activity manager analyzes the cause of the failure by evaluating the state expression. If the state fails because of a waiting document, the branch is suspended, a monitor is installed and the process proceeds with the next branch. If the state fails because of a future time-point, the branch is suspended, a monitor is installed and the evaluation process proceeds with the next branch. If the state fails because of an unknown information, the user is proposed either to monitor the state or to cancel the goal.

When no branch can be executed and when the process is waiting for events, it is suspended. At any time the user can request the suspension of the evaluation of a branch or of the activity evaluation. User suspended branches or activity evaluations can be restarted by a user resume command. The state of a process can be queried by the user at any time.

6 Built-in activities

With the activity manager, built-in activities are provided:

- Operators corresponding to classical office activities,
- Office operators as high level but still general operators,
- Procedures corresponding to standard ways of doing things in an organization.

These built-in activities are implemented by using the activity manager itself and can be used in order to built other activities.

6.1 Operators

Information handling operations can be classified into filing/retrieval related operations, communication operations and content operations.

Filing and retrieval operations apply on documents. They achieve movements of documents between the following classes of repositories:

- top-of-desk,
- temporary-repositories (table, desk),
- permanent-repositories,
- waste-paper-basket.

These movements are governed by the following principles:

- a document is uniquely identified in the system,
- a document is filed in only one place within all the permanent repositories of the office,
- a document can be present in only one place in the temporary repositories and in the top-of-desk-area.

Documents can be in two states: working document (that can be updated), and fixed document (that cannot be updated). The following considerations governs the change of state of a document.

- (i) If the document came in the office through a receive operation, it cannot be updated, it is created as fixed.
- (ii) When a document has just been created by the tenant of the office, its state is set to working until it is reset to fixed either by a user fix command or by the system itself.
- (iii) Any document of the communication type is fixed by the system at the time the document is sent. This is done to keep an image of what has been sent.
- (iv) Permanent repositories contain only fixed documents. Before filing a working document, it must be fixed.
- (v) Any document can be copied. The copy is created with a different document identifier and in a working state. Consistency of the copy with the original is not maintained.

From elementary operations, movements operations are defined. They accept incompletely defined parameters. When an incomplete definition is given, it is evaluated in the context of the operation and first completed as far as possible and then the user is requested to complete the definition. In case of improper parameter specification, the operation request is deleted. When the requested operation is not valid due to the state of the document, a choice of possible operations is proposed to the user. He can delete the request, or execute the proposed operation (e.g. send a communication document in a working state so that it can be filed) and proceed with the request, or pile the action in a things to do file and set the proposed operation in a suspended state until the piled operation is accomplished, then the requested operation will proceed.

Movement operations from or to permanent repositories are file and retrieve.

- FILE - Finds the specified document in top or temporary areas, and file it in the destination location at the following conditions: the document exists either in top or in temporary areas, the specified document is not already filed in another location in permanent area, and the document is not a working document. If the document is a working document, proposed actions are either to fix it or to send it if the document is a communication document. If the document is already in another location in the permanent area, the user is proposed to transform the operation into a move operation to the new specified location. In case of success of the filing operation, the document is removed from top and temporary areas.
- RETRIEVE - Looks for the specified document, first in top area, then in temporary area, and last in permanent area and makes it known from the top area if not already there.

Movement operations between top and temporary areas are put and get.

- PUT - Puts the document in the specified location in the temporary area at the following conditions: the document is in top area and it is not already in another location in temporary area. If the document is already in another location, a move operation is proposed to the new specified location.
- GET - Finds the specified document in the temporary area and if found, makes it known to the top area. If the document is not found, proposes a retrieve operation.

Release move destroy and recover are house-keeping operations.

- RELEASE - If the document is in top and if it is in one of temporary, permanent or basket area, the document is removed. If the document is not in top and not in temporary area, and if it is in one of permanent or basket area, the document is removed. In the contrary a choice is proposed either to destroy the document or to file the document.
- MOVE - Moves the specified document found in a location to the specified location (from temporary to temporary or from permanent to permanent).
- DESTROY - Copies the document into the basket area and removes it from all other repositories.
- RECOVER - Moves the document from the basket area to the top area.

Copy, create, and fix operations assign states to documents.

- COPY - Makes a copy of the stated document in the Top area and sets its state to working.
- CREATE - Creates the specified document type in the Top area and sets its state to working.
- FIX - Changes the state of a document from working to fixed.

Send and Receive are the two communication operations provided.

6.2 Office Operators

Office operations reflects semantic work in office (by opposition to information handling and process execution control). They are classified into information operations, processing operations and management operations.

Information operations are send operations with semantic. They are specializations of send and share the same structure. Inform is the only one presently implemented.

Inform, is a send operation to an interest list of people implying no specific request from the sender to the receivers.

of-what: collection
who: actors

Processing operations process informations within collections in the top-of-desk repository.

They are: revise, verify and evaluate.

Revise, modifies a collection using collections.

result: revised-collection

what: collection

using: collections

Verify, produces a verification report about the quality of a collection.

result: verification-report

what: collection

using: collection

applying: verification-procedure

Evaluate, produces an evaluation report about a collection.

result: evaluation-report

what: collection

using: collections

applying: procedure(s)

Management operations are related to activities control and interaction between roles. They reflect the actions corresponding to the organization in place in conformity to role distribution.

They are: control and approve.

Control, to control conformity to.

result: control-report

what: collection

using: collections

applying: procedure(s)

Approve, to approve proposed matter.

result: approval-report

what: collection

6.3 Procedures

An office procedure refers to a standard way of achieving a goal through ordered steps of actions associated to conditions. The execution of a procedure is distributed in an organization so that each actor participating in the execution needs to know only the part of the procedure describing its own contribution. Cooperation between actors is achieved through communication of documents whose structure and semantic is precisely described by the procedure and supposed to be correctly interpreted by every involved actor. Most of the existing procedures refers to a relatively limited set of actions verbs like prepare, send, request approval, request control, file, sign etc.

The procedure concept used by the office assistant application reproduces exactly this way of doing. For a procedure, a workstation has the activity network describing the part of the procedure needed by its owner. Send and receive documents operations are used in order to cooperate with other workstations. If these workstations does not have the corresponding local procedure description, the interaction is processed by their users with only limited assistance from their workstations.

As an example, a mission procedure can consist of several local procedures:

(i) request-mission for the requestor of the mission;

(ii) handle-mission-request for the secretary role including establish the mission-order, obtain approval from manager, request control from financial controller, get cash advance etc;

(iii) approve-mission-order for the manager role.

These local procedures are synchronized by documents exchanges: mission-request, mission-order, cash-advance-request.

The definition of a procedure can thus be limited to only one workstation or distributed to several workstations, in which case the workstations will be able to directly cooperate in assisting their owners in order to achieve the goal of the procedure.

In order to define a procedure, the following informations must be declared:

- definition of the informations created and used by the procedure and of their semantic descriptors;
- definition of the procedure roles, of the corresponding authority levels and of the rules allowing mapping procedure roles to actors;
- definition of the network(s) of activities and of the corresponding states describing the dynamic part of the procedure.

In the definition of the activity network, all operators, office operators and procedures already defined can be used as activities in the nodes of the network. From the activity manager point of view, there is no difference between procedures, operators and office operators, they are all activities made of networks of activities.

7 Top-level manager

7.1 Architecture

The top-level manager handles all interactions with the user at the command level and all external events. It includes:

- a top-level scheduler,
- a mail listener,
- a time-event listener,
- a mail event analyzer,
- a time event analyzer,
- a command interpreter that generates activities invocations.

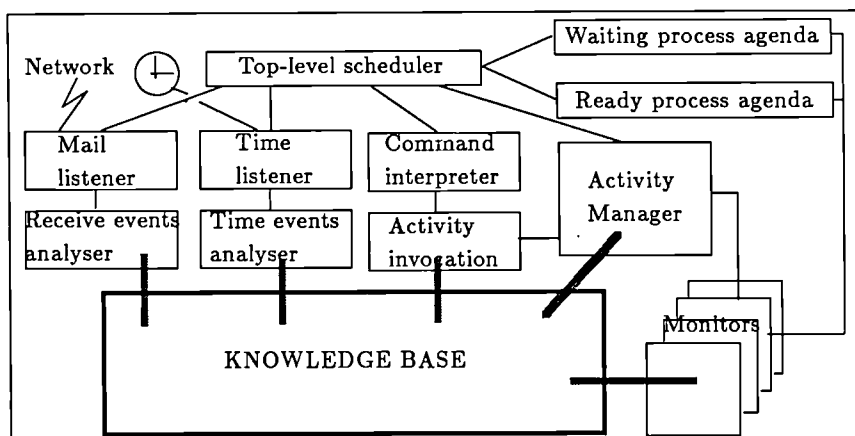


Figure 7: Top-level manager architecture

The top-level scheduler, when invoked, first activates the mail listener and the time-event listener. These modules look for either received mail or time-event and if they are found transmit them to the corresponding analyzers. Analyzers are in charge of analyzing transmitted events and to reflect the corresponding changes in the environment by updating the corresponding informations in the knowledge base. Changes made in the knowledge base activate the monitors installed by exception processing of the activity manager. These monitors put corresponding processes in the ready process agenda.

If there are no further events to be processed, the scheduler asks the user either to chose one

of the ready processes for execution, or to analyze the waiting process agenda for analyzing reasons for waiting for each waiting process, or to enter a command. When a command is chosen, the command interpreter is called and interprets the command introduced by the user. After validation, the command is transmitted to an activity invocation module in charge of activating the corresponding process.

When a process is activated its context is fetched and is transmitted to the activity manager which schedules its execution. The activity manager attempts to execute the process until either it is executed or it is suspended (by user request or because all its branches are waiting for future events). The exception processing handler of the activity manager installs monitors on the knowledge-base and things-to-do processes in the ready processes agenda when it fails to execute a branch and the user request continuation of the process. When the activity manager resumes a process execution, the control is returned to the top-level scheduler.

7.2 Incoming mail processing

All incoming communications are handled by a mail listener. The mail listener gets the documents received through the electronic mail system and transmits them to the receive event analyzer. The analyzer first attempts to recognize the received information. For that purpose, information sent by an IWS workstation are sent with a descriptor representing its internal description in a way that can be externally interpreted. The analyzer tries to interpret that descriptor.

If the descriptor is known by the workstation then the concept representing that information is created in the knowledge base, and an attempt is made to classify the information in one of the following classes: the information is an answer to a request, the information is a request corresponding to an existing command. If the information is an answer to a request, the request is updated to the received answer state and the corresponding monitor is activated (resulting in the process being put in the ready agenda). If the information is a request, the command interpreter is called and the corresponding process put in the ready process agenda.

If the descriptor is not known by the application, the analyzer put the information in a repository and inserts a thing-to-do-about-unknown-information process in the ready agenda.

7.3 Time handling

The time listener maintains a list of future time-points corresponding to future events. When activated it looks for entries in its agenda corresponding to past or present time-points and delivers these entries to the time-event-collector. The time-event-collector reflects the new situation in the knowledge base. Possibly a monitor is activated by this change and puts the process waiting for it in the ready process agenda.

The structure of time point definition enable time-point definition with precision. This can be used in order to generate time events in a natural way. A future time with precision day can be notified at the start of the day, and a future time-point with precision second can be notified in real time. This achieves a time event notification process with some primitive "common sense" approximating the one people are currently able to achieve.

8 User interface

8.1 The User Interface System

The User Interface System is built upon the X-window server of the MIT. It includes the following elements:

- the LE-LISP virtual graphic layer,
- an event manager and a virtual window manager,
- a graphical structures editor GSE,
- a FORM package.

This user interface system enables the creation of user interfaces using:

- multiwindowing
- dynamic pop-up menu
- icon interaction
- graphical designation inside network of structured objects
- and FORM presentation and data-entry.

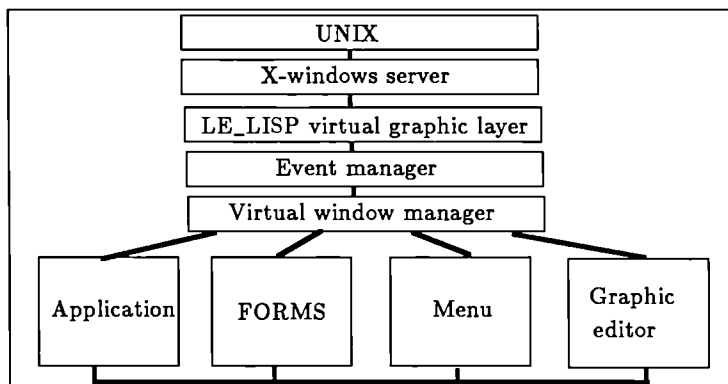


Figure 8: User Interface System

The X_window server dispatches user inputs and outputs requests from various client programs located either on the local machine or on the network. It has been interfaced to the LE-LISP virtual graphic layer by C and LISP subroutines. Titled windows and clipping area on windows have been added to X features.

On this level an Event Manager has been added. It allows, when clicking the mouse, to generate events for a specific window. The top level layer is made of a virtual window manager written in Lisp and is based on the interrupt-driven event manager. It uses a set of predefined events like `move_mouse`, `Drag_mouse`, `Click_mouse` and allows applications to define specific events for their private use. Applications can define graphical objects (area) on windows that are recognized by the event manager. The main function of the window manager is to read events from the Event Handler, determining which area lies under the location of the event, and to send a message that describes which event has occurred. The window manager allows the application to define pop-up menus associated to an area (seen as objects) in a window, the menu will be activated when the cursor is positioned in this area. An "activate" method is sent to a window when the mouse enters the window, it allows the application to define a specific behavior associated to each window.

GSE is a user interface for the presentation and manipulation of structures. It can be used to display any hierarchical structure on a bitmap screen. Direct manipulation of methods as creating and deleting graphical objects are used to simulate the manipulation of abstract objects. GSE provides a direct representation of entities in an application and powerful

semantic modeling facilities. The designer can easily specify the semantic of each object related to a class of entities and can establish relationships between these entities. Layout facilities like filtering graphic objects related to a class of objects in a window or presenting a global view of all the graphic structure in a window are provided.

8.2 User interaction

At the top level the user is proposed a choice between the following actions:

- enter a command,
- choice of a process to execute,
- show waiting events processes,
- visualize its office context (documents and their location).

The command entry starts with a command selection process implemented with hierarchical menus with the following categories:

- operators (file, send, receive, create),
- office operators (request, request approval),
- procedure invocation (buy, travel).

Then the parameters of the command are entered with a form with dialog for handling errors and the corresponding process is activated. During the process execution phase the user is kept informed of the current process context and has the possibility to suspend a branch, to suspend the process, to resume a suspended branch and to request the definition of the elements of the context.

The actions handle their own user interfaces. They mainly use the FORM package and the GSE package together with the menu utility. To choose a process in the waiting process agenda, the user is assisted and can ask for the description of each process. The same assistance level is provided for command selection.

A mechanism for easy presentation of concepts through forms has been implemented. It uses a presentation descriptor associated to each concept type describing the way it must be shown, entered or revised. This descriptor is interpreted by a module generating from it both the adequate form and the dialog to be used in order to display the concept on a form, enter the concept from the form or revise the values of the subjects of the concept that can be changed.

9 Present state and future work

From the above description, the following aspects of the prototype application have been implemented with a simple user interface:

- activity manager using monitors;
- user interface system;
- office model including organization, facilities, information and time;
- operators built as activity networks;
- simple procedure using these operators;
- the top level manager including a simplified event manager associated to monitors.

This implementation has validated the main options of the project:

- organization explicit representation,
- informations represented with their semantic,
- flexible time representation with precision,
- declarative description of procedures,
- flexible exception processing with user cooperation thanks to monitors,
- distribution of procedure description in each workstation,
- process synchronization through exchange of documents by electronic mail,
- full control of the user on the workstation operation through an adequate usage of

schedulers and agendas.

From that major step in the project, future work will consist in:

- extending the office model with more informations types and office operators,
- enhancing the scheduling process and the exception processing mechanisms,
- building a true user interface using UIMS developed by CRC with the audio capability and using the natural language facilities developed by KUN,
- experimenting the implementation of real office environment including procedure,
- implementing applications for easy description of the office context and procedures by end users.

The final prototype should be available in February 1989.

10 Acknowledgments

We first must thank the VUB AI laboratory team for their constant help in using and enhancing KRS for the project use. Whithin Bull project team, Pascal Fares implemented the top-level user interface and graphic tools, Michel Texier provided the FORMS package, Gabriel Jureidini handled the graphic object editor, Jian Zhong Li provided the activity manager, Pascal Thoraval implemented the organization model and M'Hamed Charifi designed the operators and office facilities. This work relies upon a hudge effort in actual office work analysis and modelling done by Florence Hermelin, Dominique Lestel and Dominique Hypeau.

11 References

- [1] G. BRACCHI and B. PERNICI, "The Design Requirements of Office Systems", ACM Transaction on office Information Systems, Vol. 2, No. 2 April 1984, Pages 151-170
- [2] F. BARBIC, S. CERI, P. MOSTACCI and G. BRACCI, "Modeling and Integrating Procedures in Office Information Systems Design", Rapporto interno n. 84-13, May 1984, 37 pages.
- [3] G. BRACHI and B. PERNICI, "S.O.S.: a Conceptual Model for Office Information Systems", Proc. of ACM SIGMOD Database Week, San Jose ; May 1983, p. 108-116
- [4] G. BRACCHI and B. PERNICI, "Specification of control aspects in information systems", draft, 1985
- [5] J.C. CHUPIN and V. JOLOBOFF, "A data model for office systems", BULL, centre de recherche, 1982, 23 p.
- [6] Ph. DUMAS and G. du ROURE, "Modelisation de bureau: vers un modèle événement-réponse", I.N.R.I.A., projet pilote KAYAK, Dec. 1980, 128p.
- [7] Murray S. MAZER and Frederick H. LOCHOVSKY, "Logical Routing Specification in Office Information Systems", University of Toronto
- [8] J.S. KUNIN, "OSL: an Office Specification Language",reference manual, revised February 1981, 99 pages, Massachusetts Institute of Technology
- [9] Marvin A. SIRBU Jr., Sandor R. SCOICHET, Jay S. KUNIN, Michael M. HAMMER,

Juliet B. SUTHERLAND, Craig L. ZARMER, "Office Analysis Methodology and Cases Studies", MIT publication, 1983, 199 pages

[10] Michael M. HAMMER and Jay S. KUNNIN, "Design Principles of an Office Specification Language", NCC 1980

[11] Marvin SIRBU, Sandor SCHOICHET, Jay KUNIN, Michael HAMMER, "OAM: an Office Analysis Methodology", MIT internal report, October 1980, 36 pages,

[12] Richard E. FIKES, "A Commitment-based Framework for Describing Informal Cooperative Work", XEROX Palo Alto Research Center, June 1981, revised May 1982, to appear in the *Cognitive Science Journal*, 14 pages, 8 references

[13] James H. BLAIR, "An Analysis of Organizational Productivity and the use of electronic Office Systems", inv. paper for the Annual Conference of the American Society for Information Science, July 1980, 6 pages, 12 ref.

[14] Simon GIBBS and Dionysis TSICHRITZIS, "A Data Modeling Approach for Office Information Systems", *ACM Transaction on Office Information Systems*, Vol. 1, No. 4, October 1983, P. 299-317

[15] ISO/DIS 8613/4, "Information processing - text and office systems, Office Document Architecture and Interchange Format - Part4: document profile"

[16] Ian R. Campbell-Grant, Peter J. Robison, "An introduction to ISO DIS 8613 Office Document Architecture" and its application to computer graphics.", Office Systems Division, ICL, July 1986

[17] J. A. ZAJACZKOWSKI, "An introduction to the CCITT/ISO standard on the transfer syntax and notation", British Telecom, April 1986

[18] N. NAFFAH, G. KEMPEN, J. ROHMER, L. STEELS, D. TSICHRITZIS, G. WHITE, "Intelligent Workstation in the office, state of the art and future perspectives", ESPRIT 84 status report, Amsterdam: North-Holland, 1985

[19] N. NAFFAH, G. WHITE, S. GIBBS, "Design issues of an intelligent Workstation", National Computer Conference Proceedings, 1986, p. 153-159

[20] IWS project, "Intelligent Workstation second year publishable report", CEE, March 1987

[21] E. BROWN, "Overview of a UIMS for the Intelligent Workstation", Proceedings of the ESPRIT 1986 conference, North-Holland, 1986

[22] Gerard KEMPEN, Gert ANBEEK, Peter DESSAIN, Leo KONST, Koenraad de SMEDT, "Author environments: Fifth generation text processors", Proceedings of the ESPRIT 1986 conference, Amsterdam: North-Holland, 1986

[23] Michel TUENI, "L'apport des techniques de l'intelligence artificielle dans un environnement bureautique", Actes de la convention informatique, PARIS 1986

[24] Luc STEEL, Walter Van de VELDE, Jan PAREDIS, Kris Van MARCKE, "Report on KRS formalisms", IWS deliverable D2/r2, VUB, March 1987

[25] ESPRIT project 56, "Functional Analysis of Office Requirements", Main Report, Mach 1987

[26] B. PERNICI, W. WOGEL, "An integrated approach to OIS developments (TODOS)",

ESPRIT '86 Results and achievements, Elsevier Science Publications N.V., 1987, p. 853-844

[27] Philippe DUMAS, Giulio de PETRA, Gilles CHARRONNE, "Towards a Methodology for Office Analysis", ESPRIT '86 Results and achievements, Elsevier Science Publications N.V., 1987, p. 797-811

[28] Michel TUENI, JianZhong LI, Pascal FARES, "Supporting execution and monitoring of office tasks and administrative procedures" IWS project deliverable, Bull MTS March 1987

REPRESENTATION ASPECTS OF KNOWLEDGE-BASED OFFICE SYSTEMS

K. Van Marcke, V. Jonckers and W. Daelemans

A.I.-LAB, Vrije Universiteit Brussel
 Pleinlaan 2 Building K
 1050 Brussels, Belgium

ABSTRACT

The main goal of ESPRIT project 82 is to build an intelligent office workstation (IWS). A knowledge-based approach was chosen to overcome the complexity resulting from the open-endedness of the office domain and the dynamic characteristics of the office environment. The knowledge representation system KRS is used to represent knowledge in the office domain. On the one hand, domain knowledge involves static domain concepts including concrete concepts such as persons and communication-means and more abstract concepts like organisations and roles. On the other hand, it involves a range of office activities which operate on these static domain concepts. In general, two categories of activities can be distinguished: primitive actions which directly operate on the office environment like 'send letter' or 'archive document' and higher level conceptual tasks like 'plan a trip' or 'order a book'. The execution of actions and the translation of tasks into actions must be controlled within the IWS. For this purpose, special office formalisms such as agendas, monitors and a unification mechanism were designed and implemented on top of KRS. The paper presents these office formalisms and illustrates their functionality with the description of a prototype knowledge-based office assistant and with some worked out examples of typical office-tasks.

1. INTRODUCTION

IWS (ESPRIT Project 82) is a project in the domain of intelligent office applications. The partners involved in the project are Bull (France), VUB (Belgium), KUN and OCE (the Netherlands) and CRC (Greece). The goal of the project is to build an intelligent cooperative personal workstation that assists the office-worker in the office tasks he is playing a role in.

An important aspect is the representation of office knowledge. In order to let the workstation intervene in an intelligent and intelligible way, it must know about the office-procedures that are being followed and about their states, goals and reasons. To this end, the system must be knowledgeable about the functioning of offices in general (the generic model) and about the organisation of the user in particular. At each of these levels, three types of knowledge can be distinguished: *tool-knowledge* (how to use electronic tools such as text-formatters, printers, files, telefaxes, etc.), *domain knowledge* (in the generic model: knowledge about hierarchies, approvals and signatures, communication means, deadlines, book-years, vacations etc.; in the particular model: instances of the generic concepts, organisation structure, the people that work in the organisation and their roles, goals of the organisation and its function in society etc.) and *office procedure knowledge* (the knowledge needed to interact in the office environment:

knowledge about office-procedures, tasks and activities, problems, problem-solving, planning and waiting.

The representation tool system KRS (Steels, 1986; Van Marcke, 1987) was chosen for the representation of these different types of knowledge. KRS was developed at the VUB AI-Lab within the ESPRIT project P440 (Message-Passing Architectures and Description Systems). It is based on earlier work on Representation Language Languages (Greiner, 1980). KRS is basically a flexible object-oriented language in which multiple knowledge retrieval architectures can be explicitly modeled and used. The advantage of modeling different representation formalisms explicitly in the same language is that it allows knowledge represented in different styles to cooperate. A more pragmatic advantage of using KRS is that it is implemented in ZetaLisp in such a way that it can be easily ported to other dialects, and that it runs efficiently. KRS has been successfully ported to InterLisp, Common Lisp and Le-Lisp. Porting to Le-Lisp has been done within the context of the IWS project by people at Bull, because Le-Lisp was chosen as the language to be used in the project.

The task of the VUB in the project is to design and develop knowledge representation formalisms in KRS. Those must facilitate the task of representing office knowledge (carried out by Bull, see the companion paper to this one: Ader, 1987). The present paper describes the current state of this process. More specifically, it describes three different formalisms which are currently being developed and used in our laboratory. Some of them are also used in the implementation of the activity-manager at Bull (Tueni et al., 1987). The first is the representation of priority-agendas. A priority-agenda is an agenda on which items can be placed together with a priority. The item with the highest priority on the agenda is always removed first. Multiple priority-agendas can be used to control execution of tasks and actions in a flexible and modular way. A second formalism which has been developed in KRS is a monitor formalism. Monitors allow to automatically start some activity the moment some event occurs. In an office environment, most actions are typically invoked as a reaction to some event. The third formalism enables the representation of features and the unification of feature-sets, which is among other things useful to decide between alternative solutions.

The paper first gives an overview of the aims and achievements of KRS. It is not our intention to give a detailed description of the mechanisms of the language, but rather to illustrate why it is a suitable tool for our purposes. Secondly, the three special-purpose formalisms introduced are illustrated by means of an experimental architecture simulating an office environment. This experimental architecture is being developed at VUB to test and to experiment with those formalisms. Although this office environment will not be part of the final specifications of the project, it is sophisticated enough to be a non-trivial test-bank for the formalisms we propose.

2. KRS

2.1. Philosophy

KRS is a representation tool system developed at the VUB AI-Lab. It is partly inspired by earlier work in Representation Language Languages and in particular by the language ARLO (Haase, 1986). KRS differs from popular representation tool systems like KEE and ART in that it does not provide a fixed set of formalisms. Instead it provides a language, called the concept-language, in which new or existing formalisms can be easily modeled and used by the user. It is essentially an object-oriented language featuring single inheritance, lazy construction, caching, consistency maintenance, meta-representations and explicit representation of reference.

Multiple inheritance has grown into a heavily debated controversy in the object-oriented systems community. KRS has chosen for a single inheritance mechanism with the capability to build more complicated architectures explicitly into the language when they are needed (Van Marcke, 1987). KRS objects are constructed in a lazy way, i.e. not when they are defined but when they are first used. This leads to more economic applications and more elegant definitions, especially when cross-pointing structures are involved. Caching is a technique to store computed values for re-use. This may lead (depending on the application) to serious performance upgrades. Together with caching, a consistency maintenance mechanism is needed to make sure that there remain no cached values depending on something that has already changed. All KRS objects also have a unique meta-interpreter object that contains all information about the object itself. Those meta-features are explicitly accessible and can be overridden by the KRS-user (Maes, 1987). Finally, the *referent* relation explicitly states what an object is about. This has led among others to a smooth and sound interface to LISP. A more thorough description of KRS can be found in (Steels, 1986).

KRS was originally implemented on a Symbolics LISP machine in ZetaLisp. An earlier version has been ported to INTERLISP to run on a Xerox LISP Machine. The latest version has been ported to Common Lisp by ourselves to run on for example a Sun work station. It has also been ported to Le-Lisp by Michel Tueni of Bull to run on Sun, Vax and the Metaviseur. The implementation in Le-Lisp was made in the context of this project.

2.2. The Concept-Language

The fundamental building block in the concept-language is the *concept*. A concept is a frame-like structure, grouping a set of subjects. Each subject is a named association between the concept and some other concept, which we call the filler of the subject. This way concepts are connected in a labeled network. We will describe a concept with its name (if it has one) possibly followed by a list of its subjects, like in the following example:

```
JOHN
  A PERSON
  AGE TWENTY-FIVE
  FRIEND A PERSON
    AGE (>> age)
```

The subjects are described with their name and a description of their filler. The description of this filler can be the description of this concept, or a reference to the filler of another subject. In the preceding example, the filler of the age-subject of the filler of the friend-subject of the concept with name John is described by a reference to the filler of the age-subject of that same concept ("(>> age)"). The example therefore describes a concept named John, with an age-subject with filler the concept named Twenty-Five, and with a friend-subject with filler a concept with a new age-subject with filler also the concept named Twenty-Five.

The descriptions of the concept John and of the filler of its friend-subject contain both an expression "A PERSON". This expression classifies the concepts in a strict type-tree, which is used for a prototype-style (Lieberman, 1986) inheritance. We say that the concept Person is the type of both concepts. A concept inherits all subjects from its type except those which it overrides.

The KRS concept-system is a representation language language, i.e. a language to describe representation languages. Most of the static knowledge structures are most naturally expressed directly in the concept-language. When more complicated or more dynamical architectures are needed, they can be built explicitly in the language. KRS provides libraries to facilitate this construction. Some of those libraries have already been developed and/or experimented with, such as a production-rule library, a logic-library, a cliché-library, etc. (Steels et al., 1987).

3. A PROTOTYPE OFFICE MANAGER

3.1. A General Office Model

To investigate which formalisms can be useful in the implementation of an intelligent office system and to develop and experiment with such formalisms we have developed a prototype office manager which implements a very general model of an office environment. This section first sketches this general office model and introduces the basic concepts used in its implementation. The need for dedicated formalisms to complete this implementation is briefly discussed at the end of this section. In the next section three formalisms are discussed in more detail.

In the office domain static and active knowledge components can be distinguished. Static office knowledge involves knowledge about concrete and abstract concepts in the office domain, e.g. persons, communication-means, organisations, roles, etc. Such office concepts can be modeled straightforwardly as KRS-concepts. In this approach, the KRS-concepts fulfill the role of frame-style representation structures. Modeling office concepts as KRS-concepts therefore holds the advantages commonly attributed to frame-based representation languages (Fikes 1985). Frames capture the way experts typically think about much of their knowledge. They can describe different types of domain objects, they can represent the useful relations, and they support a kind of 'definition by specialisation' which is in most domains a natural thing to do. Below we show some simple examples (in a simplified notation) of such KRS-concepts modeling a prototypical person and a specific person, respectively.

```
PERSON
A STATIC-OFFICE-CONCEPT
  NAME a person-name
  ADDRESS an address
  ELECTRONIC-MAIL-ADDRESS a mail-address

KRIS
A PERSON
  NAME A PERSON-NAME
    LAST-NAME [string "Van Marcke"]
    FIRST-NAME [string "Kris"]
  ELECTRONIC-MAIL-ADDRESS [mail-address krisvm@arti.vub.uucp]
```

Complementary, office activities which operate on these static domain concepts can be viewed as active knowledge components. Furthermore, activities like 'send a letter' or 'archive document' which correspond to concrete actions in the office domain can be distinguished from activities like 'plan a trip' or 'order a book' which are more conceptual tasks. In fact, the latter require a combination of primitive actions for their execution. Below we sketch a very general model of active office knowledge. A complex task gradually decomposes into more primitive subtasks. In the end, each primitive subtask corresponds straightforwardly to a single concrete action.

In the remainder of the text, the term *action* is reserved for a concrete, executable action in the office world while the term *task* is used to denote conceptual entities of activity. In our terminology, actions are *executed* while tasks are *handled*.

3.2. Tasks and Actions

A KRS-concept named *action* is introduced to serve as ancestor node for all KRS-concepts which model an action. As usual, subjects associate the necessary information with each action concept. In particular, the procedure which can be called to execute the action is associated with each action concept via the *execute-procedure* subject. An example is shown below.

```

ACTION
  AN ACTIVE-OFFICE-CONCEPT
  EXECUTE-PROCEDURE a procedure

SEND-ELECTRONIC-MAIL-ACTION
  AN ACTION
  SENDER a person
  RECEIVER a person
  MESSAGE a message
  EXECUTE-PROCEDURE [procedure (send-mail ...)]

```

A KRS-concept named *task* is introduced to serve as ancestor node for each KRS-concept which models a task. The task distribute-message described further on is an example for this concept. A task contains a script which is in general an analysis of the tasks in terms of more simple tasks. Such a script, called a task-handling-script, determines how the task is handled, either in terms of its subtasks or in terms of an action to be executed. A task-handling-script has a handle-procedure subject, typically filled by a lisp-procedure, that implements the functionality of the script. Libraries of general-purpose scripts can be provided and applied to handle more specific tasks.

There are two overall subcategories of task-handling-scripts. *Primitive-task-scripts* translate the task in exactly one action to be executed. We call tasks with a primitive-task-script *primitive-tasks*. *Composed-task-scripts* match a predefined combination of more basic tasks. Typical examples of composed-task-scripts are do-all-tasks, do-all-tasks-sequentially, do-some-tasks, do-best-task, etc. They play the same role as control abstractions in programming languages.

To handle a primitive task, the corresponding action must be executed. Therefore, an action is associated with each primitive-task-script via the *handle-action* subject. To handle a composed task, subtasks must be handled as determined by the semantics of the composed task. To handle a *do-all-tasks* for example, each task enumerated in the *tasks* subject of this composed-task-script must be handled.

```

TASK-HANDLING-SCRIPT
  AN OFFICE-CONCEPT
    HANDLE-PROCEDURE a procedure

PRIMITIVE-TASK-SCRIPT
  A TASK-HANDLING-SCRIPT
    HANDLE-ACTION an action
    HANDLE-PROCEDURE
      [procedure (>> execute handle-action)]

COMPOSED-TASK-SCRIPT
  A TASK-HANDLING-SCRIPT

DD-ALL-TASKS
  A COMPOSED-TASK-SCRIPT
    TASKS a task-list
    HANDLE-PROCEDURE [procedure (mapcar ...)]

```

The example below shows how a simple task is defined using a primitive-task-script.

```

TASK
  AN OFFICE-CONCEPT
    SCRIPT a composed-task

SEND-MESSAGE
  A TASK
    FROM a person
    TO a person
    MESSAGE a message
    SCRIPT A PRIMITIVE-TASK-SCRIPT
      HANDLE-ACTION A SEND-ELECTRONIC-MAIL-ACTION
        SENDER (>> from)
        RECEIVER (>> to)
        MESSAGE (>> message)

```

The next example illustrates how a complex task such as distributing a message to a set of persons is described on a very abstract level, i.e. in terms of the message to send, the person who sends the message and the list of persons who have to receive the message. The script role holds the appropriate composed-task-script. Notice that the programming cliché *transform* is used to obtain the sequence of individual send-electronic-mail tasks from the information given with the distribute-message-task.

```

DISTRIBUTE-MESSAGE
  A TASK
    FROM a person
    DISTRIBUTE-TO a person-list
    MESSAGE a message
    SCRIPT A DD-ALL-TASKS
      TASKS (A TRANSFORM
        TRANSFORM-OVER (>> distribute-to)
        TRANSFORM-WITH ?one-person
        TRANSFORMER A SEND-MESSAGE
          FROM (>> from)
          TO ?one-person
          MESSAGE (>> message)

```

It may be necessary to emphasise at this point the importance of the distinction between primitive-tasks and actions. It is true that every primitive-task translates straightforwardly to an executable action. But a primitive task is no more than a conceptual entity while an action

really operates on the environment. Therefore, dealing with failures, recovery after machine crashes, and other persistency problems are different for both kind of concepts.

From the very general model sketched so far, the need for mechanisms which control the decomposition of tasks into more primitive subtasks and the execution of the resulting actions automatically emerges. The simple examples given create the impression that modeling complex office tasks corresponds to writing a kind of high level procedures in a specialised and powerful language. But the execution of these procedures can not be controlled by a fixed interpretation mechanism. A much more flexible interpretation process is necessary to deal with the typical characteristics of an office environment such as:

- The competition for resources between different tasks and subtasks which can be reflected in differences in priority.
- Interference of externally induced changes in the office environment with the execution of tasks.
- Interference between execution of different tasks.
- The existence of alternative ways for dealing with the same problem.

The following section introduces three special purpose formalisms which are useful in the implementation of an office manager. First, *priority-agendas* are introduced and their usage to control the execution of tasks and actions with different priorities in a flexible and modular way is illustrated. Second, *monitors* are introduced and it is illustrated how they can be used to coordinate task handling. Finally, a *unification mechanism* on feature sets is proposed which can be used to deal with choices between alternatives.

4. SPECIAL-PURPOSE OFFICE FORMALISMS IN KRS

4.1. Priority Agendas

4.1.1. Basic Idea

A priority-agenda is basically a queue on which items can be stored together with a *priority-level*. The selection of the number of priority-levels is arbitrary. Upon a *first-element* request, the agenda returns the items in order of decreasing priority. Objects with the same priority are returned on a first in first out basis. In addition, the priority-levels of the objects on the agenda can be *upgraded*. For this purpose, an *upgrade-step* and a *maximum-upgrade-level* is associated with each priority agenda. Regular upgrades can for example be used to avoid that tasks with low priority-levels stay on the agenda for ever.

```
PRIORITY-AGENDA
  AGENDA a mutating-list
  UPGRADE-STEP a number
  MAXIMUM-UPGRADE-LEVEL a number
```

Priority-agendas as such are useful for many purposes. In particular, we illustrate in the following subsection how priority-agendas are used in the implementation of the prototype office manager which serves as test-bed for the development of special-purpose office formalisms.

4.1.2. The Administrative Coordinating Manager

The prototype office-manager is represented as a KRS-concept named Administrative-Coordinating-Manager (ACM). An ACM has three important components: a task-agenda, an action-agenda, and a monitor-manager. The task-agenda component is responsible for handling tasks in the office-environment, i.e. for the translation of tasks into a number of subtasks to be handled. The action-agenda component is responsible for controlling all operations on or within the office-environment, i.e. for executing actions. The monitor-manager component is responsible for feeling changes in the office-environment and for activating tasks and actions that are waiting for such events to happen. This component is described in more detail in the following section when monitors are discussed.

Both the task-agenda and the action-agenda are specialisations of the priority-agenda discussed earlier. All items on the task-agenda are tasks and all items on the action-agenda are actions. Furthermore, each agenda has associated with it an independent LISP-process that infinitely selects and 'treats' the first-element of the agenda. In the case of the action-agenda this simply means that the actions on the agenda are continuously executed. In the case of the task-agenda this means that the tasks on the agenda are handled which implies that subtasks are pushed on the task-agenda or that actions are pushed on the action-agenda as specified for the task involved. In this way, a few simple components are combined into a powerful yet flexible and modular whole.

```

ADMINISTRATIVE-COORDINATING-MANAGER
  AN OFFICE-CONCEPT
    TASK-AGENDA a priority-agenda
    ACTION-AGENDA a priority-agenda
    MONITOR-MANAGER a monitor-manager
  
```

Using priority-agendas to run tasks and actions holds several advantages over a fixed interpretation process for tasks and actions. In an office environment multiple activities are running simultaneously. The use of a priority-agenda makes it for example possible to handle urgent new tasks before completion of older, possibly already partially handled tasks. As explained earlier, to handle a complex task, subtasks are pushed on the agenda (as specified in the task's script). Therefore, another urgent task can be handled before a cumbersome but maybe not so important task is completely finished since the ACM will detect this newer and more urgent task. In combination with the monitor-manager discussed in the following subsection it becomes also possible to explicitly suspend the execution of tasks and to reactivate them automatically when appropriate.

Using two separate agendas for tasks and actions is decided upon because it further enlightens the conceptual difference between tasks and actions. In addition, using different agendas implies using different concurrent processes to execute actions and to handle tasks, respectively. This may turn out to be an advantage when for example a high-priority action is taking up much resources since part of the resources will always remain available for running the task-agenda and vice versa.

4.2. Monitors

4.2.1. Basic Idea

A *monitor* is a KRS concept which performs an action when it senses a change in the concept it monitors (the filler of its *monotoring* subject) and only if its *fire-when* subject is true at that moment. Monitors are connected to the dependency network used in the consistency maintenance system of KRS and can detect mutations to concepts that way. The *fire-action* subject of a monitor contains a form to be evaluated whenever a change is monitored. The monitor starts monitoring when the referent of its *activate* subject is computed. A monitor is deactivated when its *deactivate* subject becomes true. Once deactivated, a monitor can never fire again.

```

MONITOR
  MONITORING a concept
  FIRE-ACTION a form
  FIRE-WHEN  a boolean
  DEACTIVATE a boolean
  ACTIVATE  a switch

```

Monitors are used to implement the monitor-manager component of the ACM. This component manages the resumption of the execution of primitive and composite tasks that were waiting for events to happen. Central in the operation of the monitor-manager is the concept of a *delay* which is based on the activation of a monitor.

4.2.2. Delays

A *delay* is a concept which is used to suspend execution of tasks until something happens.

```

DELAY
  ACTIVATE
  EVENT some mutating concept
  DELAY-UNTIL a form
  DELAY-WHAT a task

```

The delay concept has subjects *event* (the concept which represents the event that is monitored for changes); *delay-what* (a primitive or composed task which is to be suspended until something happens); *delay-until* (a condition to be checked when the monitored event changes; when this condition becomes true the delay is ended and the suspended task is resumed) and *activate* (to start the delay: the *delay-what* is suspended, and a monitor is installed and activated monitoring *event*, firing when *delay-until* becomes true and deactivating itself when it has fired). The following example shows how a delay can be constructed that waits for a specific date to resume the task with which it is associated.


```

DELAY-FOR-CHANGING-DATES
  A DELAY
    EVENT (>> day of current date)

DELAY-UNTIL-DATE
  A DELAY-FOR-CHANGING-DATES
    DATE a date
    DELAY-UNTIL (>> (equal (>> date)) of current-date))

```

A *delay-for-changing-dates* checks its delay-until condition every time the day of current date changes. Its delay-until can be an arbitrary boolean condition. A *delay-until-date* has a more specific condition. It restarts its activity when it notices that the current-date is equal to the filler of its own subject *date*.

Delays can and have been used to define more complex and abstract tasks such as *monitor-deadline*, *wait-for-reply*, *wait-for-approval* etc.

4.3. Unification on Feature Sets

4.3.1. Feature Set Notations

KRS concepts describing office knowledge show a clear resemblance to a feature notation, which is one of the most conceptually clear formalisms. In a feature notation, objects and situations are described as sets of ordered pairs, where each pair consists of an attribute and a value. We will call such a set of pairs a *description*. Values of features can be atomic (i.e. a symbol) or complex (i.e. another feature). The realisation of feature formalisms in KRS is straightforward: attributes are KRS subject names, and values are subject-fillers. The expressive power of the notation can be enriched by allowing negation (represented by 'NOT') and disjunction (represented by curly braces). Notice that KRS definite descriptions can be used as values, as long as they evaluate to an atomic or a complex value. An example office concept in feature notation could be the following:

```

MY-TRIP
  TYPE TRIP
  REQUESTOR MILLIGAN
  DESTINATION PARIS
  DEPARTURE
    TYPE DATE
    DAY 22
    MONTH 6
    YEAR 1987
  DURATION
    DAYS {2 3} ;; Two OR three days
  TRAVEL-MEANS
    NOT CAR ;; Any travel means but a car
  APPROVAL-AUTHORITY
    (>> MANAGER OF REQUESTOR) ;; A definite description
  REASON
    TYPE TECHNICAL-MEETING
    WITH SELLERS
    PROJECT P-82

```

Problem solving with feature descriptions can be done through *unification*, an operation on feature sets which is very successful in current (computational) linguistics, but which can be usefully applied to other domains as well. Other operations on feature descriptions can be envisioned (e.g. generalisation; Karttunen, 1986) but we will not explore their use in office modelling here.

The type of unification which we will describe here is related to Prolog-like unification (a kind of pattern matching in which variables in the patterns matched are bound such that they become equal), but with important differences. The type of unification we use was introduced into linguistics by Kay (1979, for an introduction, see Shieber, 1986). To our knowledge, this kind of unification has not yet been applied to office problem solving.

Unification is the union of feature sets, but with two important differences: there is a possibility of failure, and unification is structure-changing; if unification succeeds, the descriptions unified are changed in the process (they are merged). The resulting description is the smallest (most general) description subsuming all descriptions which have been unified. If unsuccessful, the operands of the unification operation are left unchanged. In unifying two feature sets, only the values of those attributes which are present in both feature sets are compared. If two values are atomic, they must be equal or the unification fails. If they are complex (i.e. feature sets), the unification operation is applied recursively on them. E.g., unifying the earlier description with

```
TRIP-INFO
  TYPE TRIP
  REASON
    PROJECT {IWONL NFWO}
  REPORT OBLIGATORY
```

would result in failure, as REASON PROJECT P-82 cannot be unified with REASON PROJECT {IWONL NFWO}. However, unification of the earlier description with

```
TRIP-INFO
  TYPE TRIP
  REASON
    PROJECT {P-82 P-440}
  REPORT OPTIONAL
```

would result in the merger of both descriptions, because the unification succeeds. In this case, the result would be the addition of the feature REPORT OPTIONAL to the initial description.

The main advantages of a feature notation combined with unification are declarativeness and order-independence. Declarative descriptions are simpler to understand and nearer to human thought than procedural descriptions. Order-independence means that the order in which descriptions are unified is irrelevant (unless side-effects are allowed). This implies that the amount of control information and execution sequence information needed is small. Furthermore, as there is a direct mapping from the feature notation to the KRS concept graph (or rather a part of it), the same advantages applying to the KRS concept graph (lazy evaluation, caching, consistency maintenance and inheritance) also apply to the feature notation. The fact

that descriptions can be *named*, and used with this name in other descriptions allows modularity and conciseness in the descriptions.

4.3.2. Applications of Unification

In the office problem-solving environment, unification can be used for the control of choices among alternatives, for structure building (specialising abstract descriptions to more concrete descriptions and completing partial descriptions) and inconsistency checking. We will explain these applications in turn.

- (1) **Choice between alternatives.** Often, a choice must be made between different alternative tasks or activities. Tasks and activities may be extended with pre-conditions and post-conditions. Pre-conditions are checked against a description of the current situation. If they are true, the activity is applicable and can be executed. Post-conditions are conditions which should be true after the application of the activity. They can be used to check whether the activity was applied correctly or to prevent the execution of an activity the post-conditions of which are already satisfied. When at some point a choice must be made between two or more alternative activities or tasks, a correct choice can be made by unifying a description representing the current situation with the pre-conditions of each activity. If the unification fails, the activity cannot be executed, if it succeeds, execution is possible. Simultaneously with this unification, the description of the current situation is extended with information present in the pre-conditions of the activity. Analogously, unification of the current situation with the post-conditions of activities can be used to check whether the intended effect of an activity has been achieved.
- (2) **Structure-building.** Initially vague, incomplete and abstract (general) descriptions can be gradually provided with more detail (specialised) by unifying them with descriptions representing the office knowledge. We might start, for example, with a *travel-request* by creating a concept which simply states our intention to travel, our destination and the time-period. The office problem solving mechanism may then provide such detail as the project the applicant works on, his manager, the preferred travel means, the amount of money to be given in advance and other information necessary for obtaining approval, finding financial support, contacting a travel agent etc., by unifying the initial and intermediate descriptions with descriptions embodying office knowledge.
- (3) **Inconsistency checking.** Office work is distributed (different descriptions relating to the same task may be created by different people or at different times) and therefore prone to inconsistencies. Different descriptions can be checked for consistency by unifying them. A useful property of unification systems in this respect is that when a unification fails, the place of the inconsistency (however deeply recursively embedded) can be easily reported to the office worker.

5. CONCLUSION

We have shown that a flexible knowledge representation system and special-purpose formalisms developed for office procedure assistance are necessary prerequisites in the development of an intelligent workstation. The implementation of a prototype office environment in KRS was instrumental in the isolation of useful formalisms like priority-agendas, monitors and unification. Further experimentation with this prototype is expected to yield additional office formalisms.

6. REFERENCES

- Ader, M. and M. Tueni. 'An Office Assistant Prototype.' ESPRIT Technical Week, Brussels, September 1987.
- Fikes, R. and Kehler, T. 'The Role of Frame-Based Representation in Reasoning.' *Communications of the ACM*, Vol. 28 nr. 3 (1985)
- Greiner, R. 'RLL-1: A Representation Language Language.' Stanford Heuristic Programming Project, HPP-80-9. Stanford University, California, 1980.
- Haase, K. 'ARLO - Another Representation Language Offer' *MIT Bachelor's Thesis*, October 1986
- Karttunen, L. 'Features and Values' *Proceedings of the Tenth International Conference on Computational Linguistics*, Stanford University, Stanford California, 1986.
- Kay, M. 'Functional Grammar' in *Proceedings of the Fifth Annual Meeting of the Berkeley Linguistics Society*, Berkeley California, 1979.
- Lieberman, H. 'Using Prototypical Objects to Implement Shared Behavior in Object Oriented Systems.' in *Proceedings of OOPSLA'86*, Portland Oregon, 1986.
- Maes, P. *Computational Reflection*. Technical Report 87_2, A.I.-LAB, University of Brussels, 1987.
- Shieber, S. *An Introduction to Unification-based Approaches to Grammar*. Chicago: University of Chicago Press.
- Steels, L. 'The KRS concept system.' Technical Report 86_1, A.I.-LAB, University of Brussels, 1986.
- Steels L., W. Van de Velde, J. Paredis, K. Van Marcke and V. Jonckers. 'Report on KRS Formalisms.' IWS deliverable D2/r2, A.I.-LAB, University of Brussels, 1987.
- Tueni M., J. Z. Li and P. Fares. 'Supporting Execution and Monitoring of Office Tasks and Administrative Procedures.' IWS deliverable E/r1, Bull, 1987.
- Van Marcke, K. 'KRS manual.' University of Brussels, A.I.-MEMO 87_3, 1987.
- Van Marcke, K. 'Context Determination Through Inheritance in KRS.' University of Brussels, A.I.-MEMO 87_1, 1987.

Project No. 367

FORMATTING DOCUMENTS AND OPTIMIZATION'S PROBLEMS

AUTHOR : Jean - Michel VAN VYVE
SOBEMAP S.A., Place du Champ de Mars 5, 1050 Bruxelles
SOMIW PROJECT 367

ABSTRACT

Let us recall that a given document may be described according to two points of view : a logical one, and a layout one, and that, basically, the goal of our formatter is to allow the possibility of going from the logical point of view to the layout one.

Moreover, we want to obtain an optimized layout document.
The general notion of optimization may be shortly described as follows.

To achieve its goal, the formatter needs the notion of layout document type. Such a document type includes object type definitions, and a model of sub-types allowed for each of them.

For a specific layout document now, let us consider a function attached to a given object occurrence (even if the function is defined in the type associated to the object occurrence), having as variables the sub-object occurrences. A local optimization may be defined as the minimization of this particular function.

A global optimization will deal with several occurrences : we will minimize another function (let us denote it as $F()$) having as parameters the values returned by each function attached to the occurrences we consider.

We use a general scheme which includes the following features :

- A common skeleton is used for all functions attached to the occurrences;
- A heuristic is used to select the current set(s) of occurrences on which minimization of function $F()$ is considered.

To be a little bit more precise, this scheme uses the width, stretchability and shrinkability concepts introduced by Knuth [2] (in the very particular case of the lines). However, although it uses the similar concepts, the scheme itself is different.

Another notion used by the formatter is the constraint.

The point is that constraints must not always be satisfied : they can be relaxed, in different ways. A quality measure is associated with each of these ways.

Using these measures, a search (of a solution, that is of an optimized layout document) can be performed.

During this search, intelligent backtracking may be performed, and a heuristic analysis of the interactions between the whole set of constraints involved may also be used when necessary. However, the notion of relaxation may be used in a simpler way, provided that the layout document type was also "simple".

1. INTRODUCTION

This paper is mainly devoted to the so-called optimization problem, that is, optimization of the presentation of a document.

Straightforward approaches exist and are very common on many word processing equipments; all of these are based upon the local optimization of a single line.

In contrast, if we consider the approach of Knuth for breaking paragraphs into lines [2], we will deal with sophisticated concepts, as well as a complete search of possible hyphenation points where a line can be ended, in order to optimize a paragraph considered as a whole.

Unfortunately, this approach has an average time's complexity exponential with the number of entities considered (words or syllables).

The goal of this paper is to present an intermediate approach which :

- uses some concepts introduced by Knuth;
- generalises these;
- does not use the same scheme;
- produces better results than the local optimization does;
- has not an average time's complexity exponential with the number of entities considered;
 - uses the notion of constraint [1] .

The paper is structured as follows :

- first, the task (formatter - printer/server) is briefly described;
- second, the frame of the optimization problem (the formatter) is introduced;
- third, the problem itself (optimization) is defined, as well as the notion of constraint;
- fourth, our solution is presented, as well as a simple example to illustrate it.

2. THE TASK : FORMATTER-PRINTER/SERVER

2.1. Description of the formatter

The formatter's inputs are the following :

- a logical document occurrence, composed, for instance, of chapters, paragraphs and sub-paragraphs;
- a layout document type, which includes object type definitions, and a model of sub-types allowed for each of them. Those types may correspond to pages, blocks, lines, and so on. To put it on another way, the layout document type constitutes the type of the (future) output.
- the so-called layout directives, which are links between logical types and layout ones.

Example of layout directive : "a chapter (logical) must always begin on a new page (layout)".

The output of the formatter is a layout document occurrence, belonging to a layout document type, built from the logical document occurrence and from the layout directives.

The goal of the formatter is to produce an output according to those specifications, and to perform optimization during this process, in a way which will be precised later.

2.2. Description of the printer/server

The inputs of the printer/server are the following :

- a description of a printer device :
 - a list of high level functions with, for each of them, the following boolean information : available or not (example : rotation of characters);
 - explicit coding of low level functions (example : change the inter-character value);
- a layout document occurrence. Such a document is not necessarily a result of the formatter.

The output is a file of commands which can be used by the printer device.

Important features of the printer/server are the following :

- the automatic substitution of high level functions by low level ones, if the high level functions are not available (rotation of characters will be explicitly calculated if this function is not available as a whole);
- the building of an internal page, on which several calculations will be done before the translation in commands;
- the ability to try different possibilities in order to keep the original lines and blocks as they are in the layout document occurrence (example : change the inter-character value; if not possible change the font);
- the possibility to take account of user's desiderata (printing directives).

3. THE FRAME : THE FORMATTER

Two key notions are used in the process of formatting :

building and evaluation.

Building is performed according to the layout document type. From time to time, a given object (belonging to a layout type) has to be created (for instance when a layout directive occurs), and this resulting object must be "attached" to the existing layout structure in a consistent way, that is, with respect to the models defined in the document type. Finite automata are used to perform this job.

Evaluation. The layout type also includes some attributes, and the rules to evaluate them. The evaluation of these attributes, at a given state of the building, has been found very similar to the problem of evaluating attributes belonging to a parse tree derived from an attribute grammar. As a consequence, we will use a similar method of evaluation.

Obviously, the evaluation results depend on the building phase; similarly, the building results depend on the evaluation.

The formatting process may thus be summarized as follows :

perform the loop : (building; evaluation), until the processing has been completed.

Building phase and evaluation phase have their own properties, and induce their own problems.

However, in this paper, we are dealing with optimization, which consists of minimization of some functions (rules) belonging to some occurrences.

The optimization's problem belong to the evaluation phase (although the building phase may be affected, as we said). As a consequence, in order to focus our attention on the optimization, we will simplify the building phase, by simplifying the models belonging to the layout document type.

4. THE PROBLEM : OPTIMIZATION

4.1. Concepts

Let us consider a layout occurrence (we can call it "father") and a list of layout occurrences allowed by the model describing the father in the layout document type (we can call these occurrences "sons").

As we said, we use a very simple set of models in the layout document definition :

- some occurrences may not have any sons ("atomic" occurrences : pictures, for instance);
- the other occurrences may have any number of sons (including zero).

We assume the existence of an attribute occurrence belonging to "father" (denoted as "K"), which can be calculated as a function of attribute occurrences (denoted as "ki") belonging to the sons.

We have : $K = f(k_i)$

We also assume the existence of a function of "K", denoted as $p(K)$, with the following properties :

$p(K) > 0$;
 $p(K)$ and $p'(K)$ exist and are continuous for all K;
 $K=0 \Rightarrow p(K)$ is minimum, and it is the only one;
 $K>0 \Rightarrow p(K)$ is increasing with K;
 $K<0 \Rightarrow p(K)$ is decreasing with K.

Such a function will be called a penalty.

It is extremely important to realize that these concepts may be used between an arbitrary father and its sons.

However, if the father is a line, the classical concept of width, stretchability and shrinkability introduced by Knuth [2] may be used; such a case is not general one.

4.2. Optimization's taxonomy

Local optimization : adding available sons (as we said, we deliberately skip the problem of building them) until the penalty function decreases; then, removing the last one. We can call a father locally optimized as "full".

Best optimization : putting sons into the fathers in such a way that :

- The order in which the sons appear does not change, despite the fact that they belong to one father or another;
- a function $F()$ of the penalty of all the father is minimized.
(Remark : if the penalty is carefully chosen, this function $F()$ can be reduced to a sum).

To put it in another way :

local optimisation : search for a local minimum;
best optimisation : search for the global minimum.

It is clear that the average time's complexity of the best optimization method is exponential : the whole set of possible configurations has to be tried.

Global optimization : a method which :

- has not such a time's complexity;
- gives better results than the local optimization.

"Better results" meaning that :

the value returned by the function $F()$ (sum) of the penalty of all fathers, using a global optimization

is little that

the value returned by the function $F()$ (sum) of the penalty of all fathers, using the local optimization.

4.3. Constraints

4.3.1. Definitions

Before the description of the global optimization we have chosen, let us introduce the concept of constraint [1].

First, we have an equation :

$$f(\text{Name1}, \dots, \text{NameK}) == \text{TRUE}$$

when $f()$ is a logical expression, and $\text{Name1}, \dots, \text{NameK}$ attribute occurrence names.

At this point, we would like to have some more flexible notion than an equation which returns true or false : the notion of "measure of the quality" of a constraint has to be introduced here.

We have thus the so called constraint's utility :

$$U = U(f(\text{name1}, \dots, \text{namek}), \text{name1}, \dots, \text{namek});$$

which returns a real number.

The connection between both notions is the following :

$$f(\text{name1}, \dots, \text{namek}) == \text{TRUE}$$

↓

The constraint is satisfied

↓

The utility is maximum.

The central idea is that constraint's satisfaction is not mandatory. In such a case :

$$f(\text{name1}, \dots, \text{namek}) == \text{FALSE}$$

↓

The constraint is unsatisfied

↓

The constraint may be relaxed, if allowed to be.

We urgently need the definition of what a "relaxation" is.

Relaxation's Definition :

Substitution of the value of one or several of the $\text{name1}, \dots, \text{namek}$ variables by other values taken from lists which are part of the definition of the constraint

or

substitution of the whole equation by another one.

A utility value is associated to each possible substitution (relaxation). This is the reason why the equation itself is a parameter of the utility. We have the following interpretation :

the higher the utility, the best the relaxation.

Now, in general, relaxation and their associated utility may be used to rate a search space of states. During this process, heuristics and intelligent backtracking may be used.

Nevertheless, it is quite obvious that the constraint's formalism will be more simple for particular constraints.

An obvious constraint can be a limitation of the range of the parameter "K" introduced in 4.1.

We can have :

$$A1 < K < A2$$

and a possible relaxation should be :

$$B1 < K < B2,$$

with $B1 < A1$ and $A2 < B2$.

4. GLOBAL OPTIMIZATION : A SOLUTION

Let us consider a text currently under processing by the formatter.

We assume :

- being in the frame of the following layout document type :

- document : any number of lines;
- line : any number of words.

This document type is only an example which has been chosen for presentation purposes.

- that the last line is locally optimized ("full");
- that the other ones are already globally optimized (we call them "balanced").

The situation is the following :

	<u>lines</u> :	<u>number</u> :	<u>penalty</u> :
<balanced lines>			
<balanced>	under processing by	i	P _{1i}
<full>	the formatter. We	j	P _{1j}

We can denote this situation as "S1".

We are going to try to decrease the sum of the penalties of the two last lines.

First, we put the first word of the last line at the end of the previous one :

	<u>lines</u> :	<u>number</u> :	<u>penalty</u> :
	<balanced lines>		
<no more balanced>	under processing by the	i	
<no more full>	formatter. We	j	

At this point, we have to restore the property of the last line (to be full).

After that, we will have :

	<u>lines</u> :	<u>number</u> :	<u>penalty</u> :
	<balanced lines>		
<no more balanced>	under processing by the	i	P2i
<full again>	formatter. We assume	j	P2j

We denote this situation as "S2".

Similarly, being in the situation "S1", we can put the last word of the previous line at the beginning of the last line, then possibly remove some words in order to make it full.

We will obtain :

	<u>lines</u> :	<u>number</u> :	<u>penalty</u> :
	<balanced lines>		
<no more balanced>	under processing	i	P3i
<full again>	by the formatter.	j	P3j

And we denote it as "S3".

At this point, we are able to compare the following sums (assuming that we combine penalties by summing them) :

$$\begin{aligned} P1i + P1j \\ P2i + P2j \\ P3i + P3j \end{aligned}$$

The interesting situation occurs when $P1i + P1j$ is not the minimum of the sums.

Let us assume that $P2i + P2j$ is the minimum. It means that "S2" is the best situation :

	<u>lines</u> :	<u>number</u> :	<u>penalty</u> :
	<balanced lines>		
<no more balanced>	under processing by the	i	P2i
<full again>	formatter. We assume	j	P2j

Shortly speaking, we have improved the quality of the two last lines, having payed the price of destroying the "balanced property" of one of them. Clearly, we cannot let our set of lines in such an odd situation.

Let us denote the last line as " $l + 1$ ", the previous one as " l ", and so on, and the process of building situations S_1 , S_2 , S_3 and selecting the best one as "balanced operation". What we have done was :

balanced operation between " l " and " $l + 1$ ".

Let us precise that "making last line full again" is obviously specific to the last line : we will not do it if we perform a "balance operation" on other lines.

At this point, it becomes clear that, in the general case, if a balance operation occurs between line l and line $l + 1$, and if its result is the selection of an " S_2 " or " S_3 " situation, both line l and line $l + 1$ will become unbalanced, and we will perform two other balance operations :

Between " $l - 1$ " and " l ";

Between line " $l + 1$ " and " $l + 2$ ".

The central question is now : when will this process stop ? When a " S_1 " situation is selected; or if the very first line or the last one is involved.

Practical tests on our formatter have shown that this kind of recursive calls stops very fast; in fact, more than single calls occur rarely.

As a final remark, let us insist on the fact that the concepts of :

- global optimization;
- full property;
- balance property;
- balance operation;

are general ones. The presentation in terms of words and of lines has been chosen to allow a kind of "visualization" of the process.

Generally speaking, those concepts are derived from functions defined in the layout document type.

REFERENCES

1. FOX, M. S., Constraint-Directed Search : A case Study of Job Shop Scheduling, PhD thesis, Carnegie-Mellon University, 1983.
2. KNUTH, D. E., and PLASS, M.F., Breaking Paragraphs into lines, Stanford Department of Computer Science, Report No, STAN-CS-80-828, Nov. 1980.

Project No. 956

COCOS architecture and its MMI model

Najah NAFFAH
Michel TEXIER

BULL MTS
Direction des Etudes Avancées
7 rue Ampère, 91343 MASSY Cédex, FRANCE

Abstract

The COCOS (COmponents for future COmputing Systems) ESPRIT project goal is to define the components of a total system architecture, from applications to the silicon, aimed at all types of users: system designers, programmers, end users.

This paper is divided in two parts. In the first part, we describe the COCOS approach for defining a global system architecture, and in the second part we focus on the topic of MMI. Other papers in this proceedings [1] [2] give details on various scenarios and tools adopted in COCOS.

In the MMI component, three layers of interface development tools are identified: the basic graphic layer (e.g. windowing server), the mid-level tools layer (eg. toolkit for creating menus or managing scrolling) and the high-level tools layer (eg. complex forms or tree editor).

To provide the greatest flexibility in interface display and manipulation, all three layers elements need to be configurable, which leads to the definition of a MMI model describing them, along with the global MMI concepts, to facilitate interface description and promote uniformity.

We believe that such a model can provide the basis for developing complex applications in the end-user environment and become a step towards defining a general UIMS (User Interface Management System).

1. COCOS Architecture

1.1. General description of the COCOS architecture

As stated in the goals definition [3], the main idea behind COCOS is to provide a set of common tools for building future computing systems. Hardware and software issues will be combined in a unique design that is based on a top-down approach. We will start by defining the functional view at the application (or user) level. This will determine the basic requirements. The next level will be the components such as operating systems and languages compilers, and their interactions. At the

lowest level, we find the VLSI, which best support the second level. This is summed up in figure 1.

In the architecture that has been defined, the following choices have been made:

1.1.1. Application level

At the application level, two categories of users have been considered: the end user taken in an office environment, and the programmer.

For the end user, a dynamic system is provided, where the user can define his tasks and the procedures he would like to apply in his decision process. This is done with the help of high level interface tools which are defined in the MMI part.

For the programmer, a proposal for a new language, called Le - Tool [2], which is object oriented and strongly typed, is defined. Productivity and reliability are the two important goals.

For both classes of users, it has been recognized that there is a need to communicate information about objects and to describe large taxonomies of objects for building and maintaining systems. Other requirements are to simulate objects and produce executable forms of objects. In order to reach these goals, Protégé has been proposed. It is constituted of a library of objects which can be used as the "raw material" when defining new systems.

1.1.2. Middle level

This level is based on GOM, a General Object Model to be built. It has been admitted that an ideal object model requires features that do not necessarily exist in one programming language. GOM must be general enough to encompass the features of many languages.

In order to define the model, we have selected a prototyping language called Parlog [4]. This language has its roots in Prolog, while adopting a model of communication known as CSP (Communication Sequential Processes) developed by T. HOARE. Parlog has been developed at Imperial College in London. It was felt that it provides the adequate mechanisms to model the interactions of complex systems and control these interactions. Another approach was to build on top of the Lisp language. The discussion has resulted in the study of some object oriented extensions to be added to Lisp.

1.1.3. Low level

The low level deals with hardware. The design proposed by the engine group is to create on top of the background of one partner and avoid building around new chips. Although other approaches have been explored (Prolog engine and parallel architecture), the group decided to combine a general purpose CISC chip (68020) and the micro-programmable symbolic chip called μ sync. This engine is to be designed for executing the Sequential Parlog Machine (SPM). This is described in [5].

1.2. Project organization

COCOS covers areas such as Hardware (standard CPU or CISC chips, RISC chips, internal and external bus architecture, parallel architecture, interconnection schemes in multiprocessor systems), Software (operating systems, distributed systems, programming languages, object oriented environments, third generation and fifth generation languages in a heterogeneous environment), Man-Machine Interface (windows, forms, selection devices, ...).

In order to investigate all these topics, and extract the basic principles from the current research done internationally, the project management team has decided to organize different working groups, and assign to each group a subject for which a state of the art report has been produced and a goals definition has been presented. All this information has been evaluated during the working meetings. It generated a global scheme for Software and Hardware design. A very intensive program of experimentation and evaluations has been defined for the first two years: 1986 and 1987.

The working groups are organized as follows (the name of the leading partner for each group is

underlined):

- 1. Model BULL - ICL - OLIVETTI
- 2. Language INRIA - BULL - ICL
- 3. User Interface ICL - BULL
- 4. Distributed system ICL
- 5. Development environment ICL
- 6. Engine BULL - SGS - ICL - INRIA - NIXDORF
- 7. Peripheral technology BULL - SGS

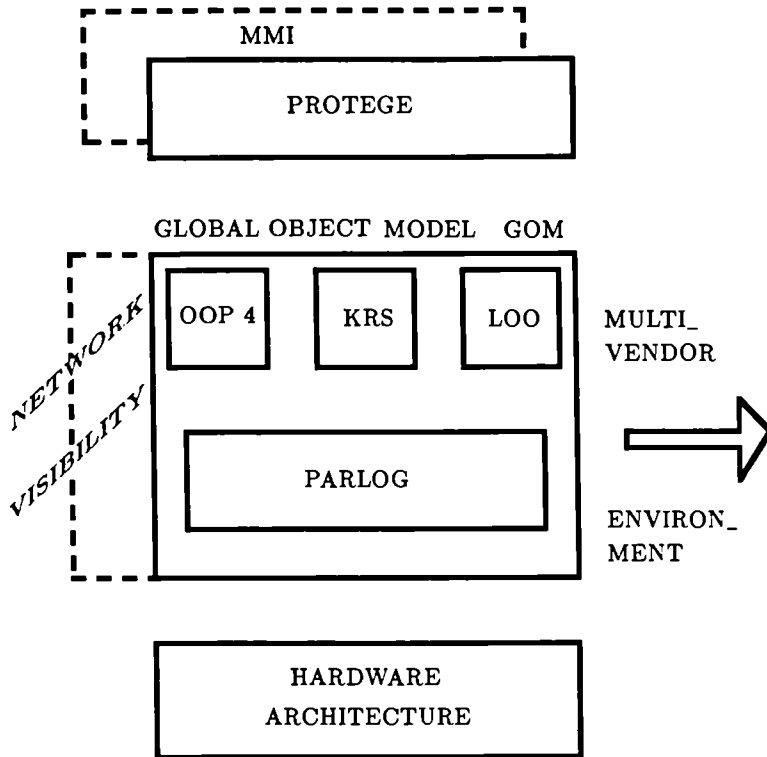


FIG.1 WORKING TASKS DEFINED ACCORDING TO THE COMPONENTS

1.3. First software scenario: Parlog

Three software scenarios have been identified as shown in figure 2. We give here a description of the first scenario, where we will focus only on the language aspect. The hardware is described in [1]. In [6], a detailed review of the Le - Tool scenario is presented.

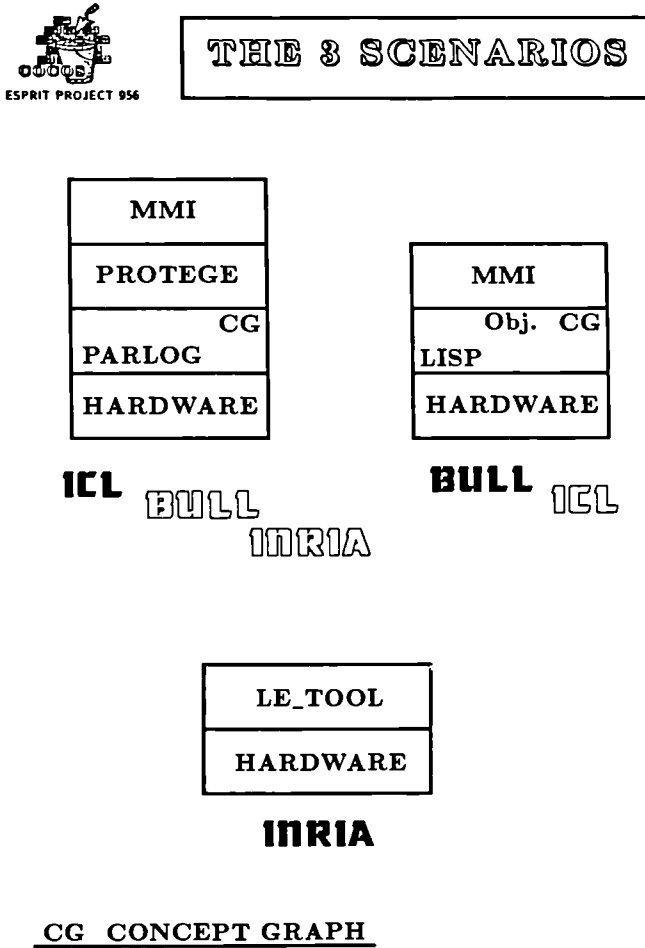


Figure 2

1.3.1. Description of the Parlog language

Parlog is a logic programming language in the sense that nearly every definition and query can be read as a sentence of predicate logic. It differs from Prolog in incorporating parallel modes of evaluation. For reasons of efficient implementation, it distinguishes and separates and-parallel and or-parallel evaluation.

Parlog relations are divided into two types: single-solution relations and all-solutions relations. A conjunction of single-solution relation calls can be evaluated in parallel with shared variables acting as communication channels for the passing of partial bindings. Only one solution to each call is computed, using committed choice non-determinism. A conjunction of all-solutions relation calls is evaluated without communication of partial bindings, but all the solutions may be found by an or-parallel exploration of the different evaluation paths. A set constructor provides the main interface between single-solution relations and all-solution relations.

1.3.2. How Parlog has been found as the target language for COCOS [7]

In COCOS, there is a declared aim to provide support for both third and fifth generation languages. We have been looking specifically at committed choice logic languages as the most promising of the currently proposed fifth generation languages.

1.3.2.1. Initial languages assessment

We looked at both the functional and the logic schools of declarative languages. We were interested specifically in the flexibility of expression. We looked at the new pure functional languages as represented by Hope and Miranda, the logic language Prolog, and the committed choice languages such as Parlog, Concurrent Prolog and GHC.

In functional languages, there is no satisfactory way to maintain local state information. It is also extremely difficult to describe interactive systems. Such systems that have been described have been small and not suitable for scaling to larger systems.

As for Prolog, we felt that it is only suitable for the development of small systems. Prolog had an advantage over the pure functional languages in that it is possible to represent state information in the language.

1.3.2.2. Parlog

The committed choice logic language which we based our analysis on was Parlog. Parlog is being developed at Imperial College, London, and was a convenient choice for us since one of the partners already had existing links with the Parlog group there.

At present there are implementations of three committed choice logic languages: Parlog, FCP, and FGHC. After careful examination of the features of these three languages, it is clear that they are basically very similar. We do however feel that the Parlog language is the most general of the three. This fact is not obvious initially because Parlog has traditionally been presented on its model form as a compiler oriented language. We have since abandoned the model form of Parlog and now concentrate on the more general Kernel Parlog.

It is clear enough that each language requires certain enhancements to bring each to the most general level. Parlog requires improved unification primitives, while FGHC and FCP require control meta-calls and additional clause and goal conjunctions.

Stereotypical predicates

In order to develop our knowledge of Parlog, we developed our object model. We found that it was possible to produce the sort of behaviour we were after by connecting together networks of lightweight processes which together appeared as a larger heavyweight process.

We called stereotypical predicates the predicates which defined these networks of lightweight processes.

The ease at which we had arrived at a network to produce the required behaviour has given us confidence that similar stereotypical predicates can be developed to produce the behaviour of more complex systems. We intend to use these techniques to implement conceptual structures.

1.3.3. the Parlog compiler

The current compiler from Imperial College is unsophisticated and attempts few optimizations of the resultant code. Parlog is a tractable language which is susceptible to many optimizations which are described in [1].

2. The MMI model

2.1. What is MMI

Man Machine Interface issues have taken an increasing importance in the last years, due to the emerging of a new and important class of non expert computer users. The need for user-friendly interfaces for these users associated to the progress in technology allowing reasonable cost bitmap screens have originated a lot of research and developments around graphical interfaces, from multi-windowing systems and toolboxes to application frameworks and User Interface Management Systems. The last ones are motivated by the development cost of user interfaces and the high level of expertise required from developers due to the complexity and richness of graphical tools.

In such a context, we propose a layered MMI architecture which goal is to encompass all issues relevant to graphical interfaces and provide the environment to classify and integrate graphical MMI tools and systems.

In the next section, we give details about the goals, requirements and state of the art leading to the proposed architecture. Then we describe the principles of the framework for organizing MMI components (the three layers, the MMI model, the presentation aspect) and how it addresses various MMI issues and requirements, followed by a presentation of the ongoing research and completed developments.

2.2. Goals, requirements, state of the art

We started working on MMI in the context of two other ESPRIT projects that have been launched in 1984: SOMIW [8] for Secure Open Multimedia Integrated Workstation and IWS [9] [10] for Intelligent WorkStation. The first project focuses on multimedia integration, e.g. voice input and voice commands mixed with still images and motion pictures. The second project major concern was end-user computing of AI-based expert systems for the office. From these two projects, we have identified a specific set of requirements that have been added to those of COCOS.

2.2.1. IWS needs in MMI

Inside the IWS project, one of the tasks was to develop an Activity Manager System (AMS) [11] which purpose is to model office tasks and allow a knowledge engineer to create an application by organizing tasks in activity networks. The need to perform this work interactively and in a user-friendly manner (for creating networks, updating knowledge bases, generate the applications end-users interfaces usually based on menus and simulations of actual paper forms, interaction screens, control panels, sets of icons etc...) calls for high level interface tools such as versatile forms, tree or icons editors on top of a graphical server and toolbox. Such tools have been developed and are presented in 2.3.2.5.

2.2.2. COCOS goals in MMI

The COCOS (COmponents for future COmputing Systems) project goal is to define the components

of a total system architecture, from applications to the silicon, aimed at all types of users: system designers, programmers, end users.

Because of this goal generality, it didn't seem relevant to us to tackle very high level issues such as the design of an application interface generator, such as a UIMS, which are more reasonable to address in a more restricted application environment.

Furthermore, we believe that "clean" environments for the development of such systems have not been developed yet, and our generality goal leads us to consider these systems as applications in COCOS. It then became natural to address the issue of identifying a general framework for organizing MMI components capable of including all levels of MMI tools and offering a MMI model encompassing these levels.

2.2.3. Developers requirements

Along with interface development tools, and to support them, developers need to describe the interface objects which are to be manipulated, from windows and simple graphical objects to menus to forms, panels, interaction screens, trees, graphs, networks etc...

A semantic meaning needs to be associated with these objects, through procedural attachment for instance. A knowledge based environment is the natural answer to these requirements.

2.2.4. End users requirements

End users are generally not computer specialists. They need user-friendly interfaces, which implies (non exhaustively) the following features:

- visual and direct manipulation of objects simulating real world objects or clearly representing the users usual tasks objects and actions.
- instantaneous feedback: any user command needs such a feedback, in order to make him accept the system and avoid wrong reactions and erroneous conclusions (the system is down, the command was wrong ...).
- execution monitoring: users often require the power to switch from one task to the other, suspend, resume a task (allowing, for instance, to consult data elsewhere which are useful for continuing the task).
- adaptability and evolutivity of the interface: users range from novice to expert, and their expertise may dramatically improve in a few sessions with a given system. The application interface should then adapt itself to the user level of expertise or at least offer a number of interaction levels.

2.2.5. State of the art

Today most of the products proposing an environment for the development of graphical interfaces are graphical servers, i.e. multi-windowing systems with a graphic library [12]. They usually offer a powerful set of basic graphical functions (screen multiplexing through overlapping windows, manipulation of the windows, drawing and filling, clipping regions, events ...) and even higher level ones (menus, text input, rubber bands ...). Some of these provide a bundled solution where high level tools and basic functions are integrated: Mactools and Windows. Others are based on the server model: X and News. X11 [12] [13] client/server approach, portability, good response time, clean and rich C interface make it a good candidate for a standard and suitable support for our MMI architecture, although it is not as versatile as News [14].

At the other end of the spectrum, much research is being done in the area of UIMSs. We don't believe these systems can be practical for the time being because their application domain is too restricted. The reason for this is that the environments (i.e. graphical servers) they rely on are of a too low level and it is too much work to implement a complete underlying knowledge based environment for all MMI components. Furthermore it would be quite redundant to recreate it for each system.

A solution could be to develop these UIMSs on top of intermediate environments proposed above some graphical servers, i.e. toolkits such as Xtoolkit [15] or application frameworks such as MacApp [16] [17]. The problem with such environments is that they are strongly influenced by the philosophy and features of the servers they are built on. This implies limitations with the addressed interface concepts (e.g.: the mouse has one button for MacApp, an interaction object is a window for Xtoolkit) and a number of other potential interface elements are not dealt with (such as editors).

We thus believe there is a gap to fill between graphical servers and UIMSs, and our goal in COCOS is to make a proposal to reach such a goal, and implement a prototype of this architecture. Next

chapter thus presents our proposal and describes the elements which have been implemented.

2.3. A framework for organizing MMI components

2.3.1. The architecture principles

These principles are: a layered architecture and configurable MMI components leading to a MMI model. They are summed up in figure 3.

2.3.1.1. A layered architecture

Our experience in developing graphical user interfaces has lead us to identify three layers of MMI tools:

- the low-level layer encompassing the basic primitives for window manipulation, basic input (keyboard, mouse, network, clock) and output (display of graphics in various modes) events, clipping, ...
- the mid-level layer offering the management of general tools developed with those of the low-level layer: selection/dragging/activation of objects (eg. buttons), rubber bands, scrolling, menus, text or graphics editing, tree display etc...
- the high-level layer for the "full scale" tools, i.e. which use, combined or stand alone, can be sufficient to build some applications interfaces since they provide the means to link tool actions to applications functions. These tools address a specific type of interface need (e.g.: managing hierarchies or covering all types of input/output needs in a window). Examples of such tools can be a tree editor allowing interactive tree building and the possibility to associate a semantic meaning to such an action, a versatile forms system encompassing needs as diverse as simulating paper forms or interactively create control panels and interaction screens, an intelligent icon manager allowing graphical and semantic dependencies between any MMI components.

2.3.1.2. The MMI model

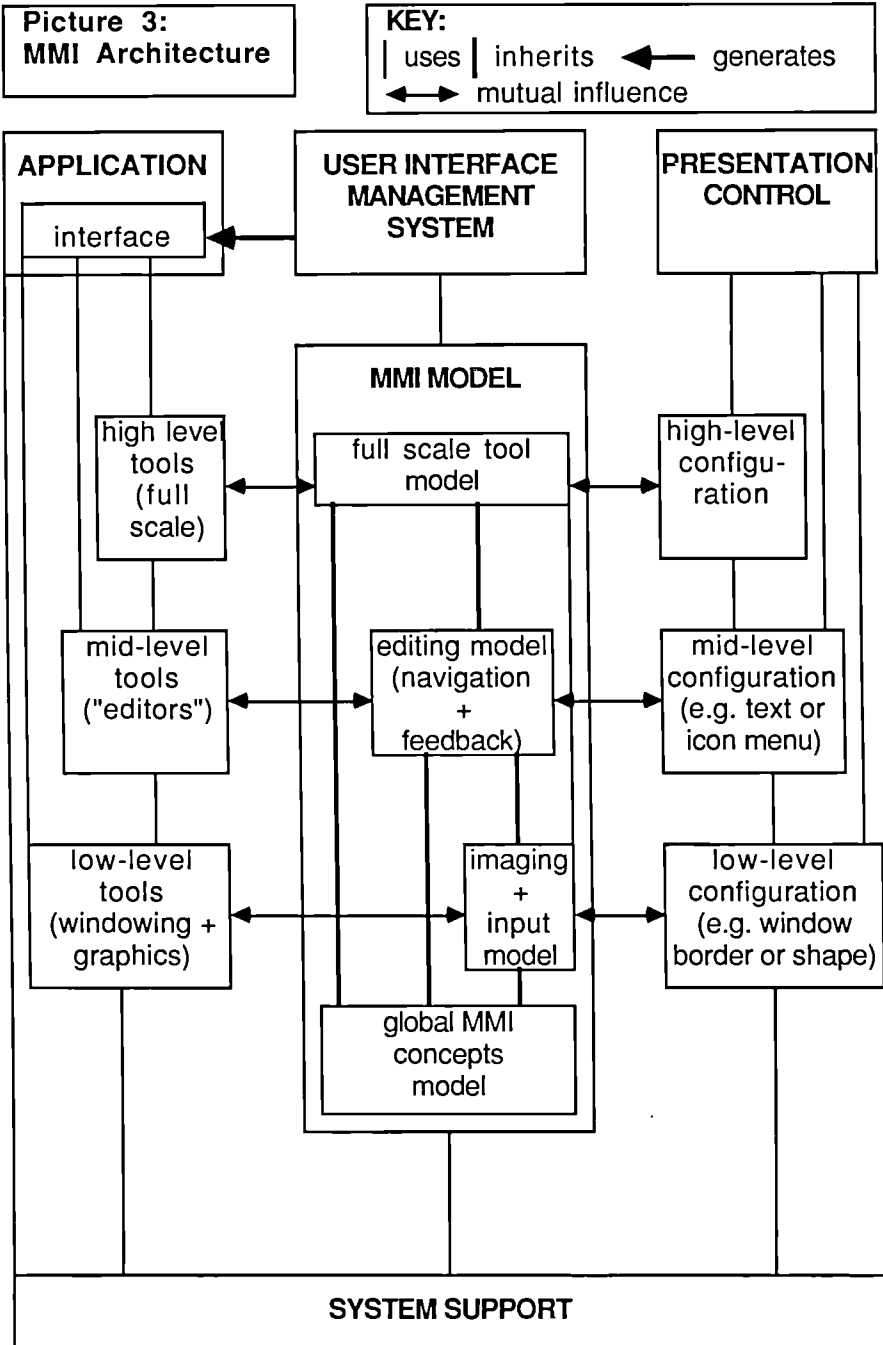
All three level MMI components should be configurable in order to cover the largest spectrum of MMI needs and allow interface customization. For instance, windows may have various shapes, colors, border and title bar aspects; dragging could be materialized by showing an actual picture moving or only a shadow; menus may include text, graphical objects, bitmaps, icons, be fixed or pop-up, blocking or not any input other than an item selection; trees can be displayed horizontally or vertically, be materialized through nodes and links or simple text indentation etc...

Furthermore, a MMI tool of a given level usually uses elements of the underlying layers: for example, forms and tree editors can be used through menus and manipulate graphical objects to build a form skeleton element or a tree node, menus are displayed in windows and their items can be graphical objects, etc...

Finally, we have developers requirements and UIMSs needs of MMI elements description and semantic association to them.

All this clearly calls for the design of a knowledge oriented MMI model encompassing the description of the general MMI concepts, such as dialog, input, output, navigation, feedback, and the classes of MMI objects manipulated, some universal (usually of a low level) such as windows, rectangles, ellipses, lines, bitmaps, menus, some depending from application needs (usually of a high level) such as forms, tree, intelligent icons editors.

The various layers tools implementation will thus rely on the MMI model describing the interface concepts and objects. The tools configurability, evolutivity and more generally power will then depend on the model organization. For instance, if in the model KB(s), the Window object is a specialization of an object called Screen-Object or Graphical-Object, it will allow windows to have various shapes and be manipulated by a structured graphics editor, which won't be the case if they are considered as rectangles or frames.



2.3.2. Implementation

2.3.2.1. Supporting environment

As evoked in 2.5., X11 has been selected as the graphical server. The X window system is a network transparent windowing system providing high-level graphics to a hierarchy of overlapping and resizable (sub)windows. The server network transparency and device independence is due to the network protocol definition of the X base system: asynchronous stream-based inter-process communication replaces the procedure call or kernel call interface of most window systems [13]. In addition, we intend to assess the Xtoolkit library package. Xtoolkit is a MMI toolkit layered on top of X proposing a philosophy for describing interaction objects (or "widgets"). These widgets feature resources, a behaviour and a presentation. The initial library of widgets includes objects such as labels, radio buttons and menus.

Le_Lisp 15.2 [18] has been first chosen as the development language for two reasons:
 - it includes a simple and clean object oriented language providing (multiple) inheritance,
 - because one of its goals is full portability, a "virtual bitmap" environment has been designed within Le_Lisp, which provides a set of primitives for managing windows and graphics. This environment has been carefully designed to encompass most of the potential needs, and should a new one occur, it is easy to develop the necessary primitives using the X11 C interface.

The two other languages, Parlog and Le-Tool, that are explored in COCOS, will be assessed as potential support for the MMI architecture.

The natural concurrency of Parlog [3] allows to associate event streams to screen objects that can be monitored by one or more Parlog processes. This should be quite interesting for managing synchronization and complex interactions between objects as communication between lightweight concurrent processes, but it may not be possible to reach adequate performance levels with today machines.

Le-Tool, as explained in [2], is a typed object oriented language, and it is planned to provide a user-friendly (graphical) syntactic editor. It thus may provide natural facilities for constraining some (it is not compulsory to type an object or a slot) of the interface objects features and associate a behaviour to these objects.

2.3.2.2. The MMI model

At the current stage of the design process, the MMI model is viewed as a set of Knowledge Bases. Figure 4 shows parts of such KBs.

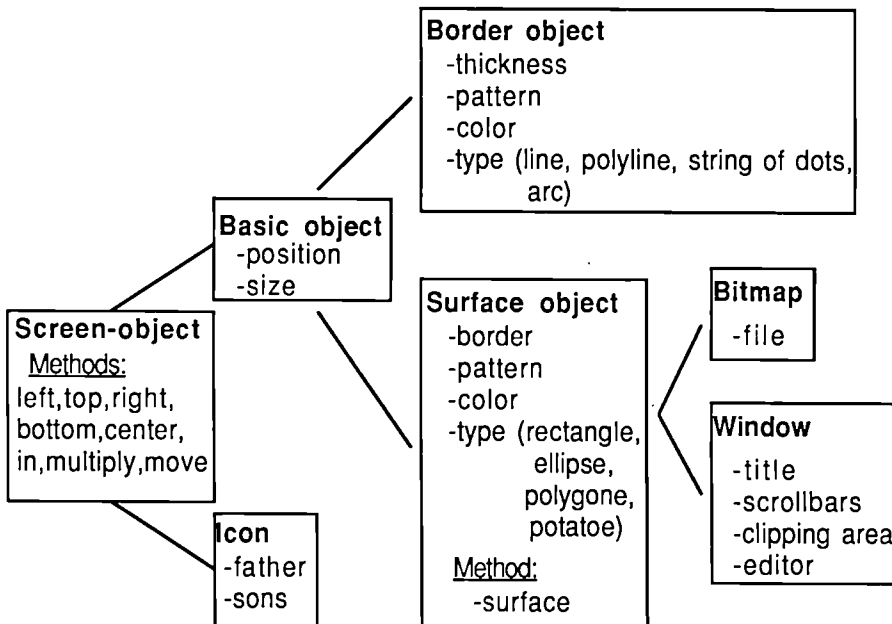
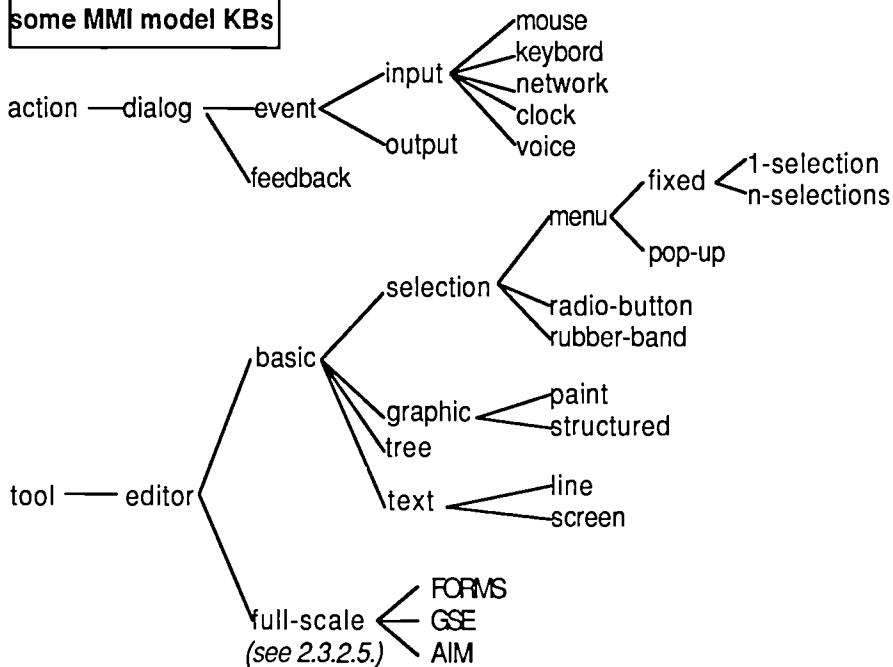
The first KB, or ACTIONS KB, encompasses global MMI interaction concepts such as dialog or feedback.

The second KB, or TOOLS KB, features the MMI tools (for the time being, the editors). The full scale editors are the tools of the MMI architecture high level layer defined in 2.3.1.1., examples of which are described in 2.3.2.5.

Finally, the third KB, or SCREEN-OBJECTS KB, presents some of the MMI objects displayed on a bitmap screen. A number of these objects slots (i.e. presentation features) and methods (i.e. messages they can react to) are shown. For the Border-object and Surface-object classes, the set of possible values for the type slot is also given.

Before implementing the completed MMI model, we will study if it is possible to map it on Xtoolkit (problems may occur with its window approach for the widgets).

Picture 4:
some MMI model KBs



2.3.2.3. The MMI architecture first layer

This layer is implemented on top of the (extended) `Le_Lisp` virtual bitmap environment (called `Virbitmap`), which provides the basic primitives for managing windows, displaying graphics, and reading input events. The `Lisp` toplevel has been redefined to behave as an event filter.

Most of the basic screen objects have been implemented, that is to say lines, polylines, rectangles, ellipses, strings, bitmaps. Messages they can receive are: left, top, right, bottom, center, surrounding rectangle, in, draw, fill, invert, erase, move, multiply, intersect etc...

The case of windows and events which are predefined structures in `Virbitmap` remains to be studied for later integration in the MMI model.

2.3.2.4. The MMI architecture second layer

A set of basic navigation functions and editors have been developed at this layer. The navigation functions allow to move the screen objects defined at the first layer. The editors include rubber-bands, box checking, and a menu editor (for any objects scattered in a window) for the selection type, a structured graphics editor for the graphic type, tree editing, and various line and screen text editors.

2.3.2.5. The MMI architecture third layer

Three tools have been developed so far at this level, all of them relating to a wide range of application needs.

The FORMS system

The FORMS system [19] goal is to cover various types of interface needs for input and display of information through windows, such as actual paper forms simulation, control panels, dialog boxes, interaction screens for spreadsheets and data bases etc...

In a form, materialized by one or more pages, i.e. windows, we distinguish between the variable information to be entered by the user or mapping some application data, and the fixed information. The fixed information makes the form skeleton where we distinguish again between the décor elements, such as logos and information on what is displayed and how to fill, and filling areas allowing editing of the variable information and communication between the form interface and the application engine through procedural attachment, in order to associate actions and feedback with user input.

A filling area can be a full page or a sub-area of it, called a part. It can be made of décor elements (first layer objects such as text, bitmaps, graphical objects) and recursively of sub-parts. It holds information about editing (in particular, the associated editor from the architecture second layer such as line or screen text edit, structured graphics edit, menu selection, box checking) and interface to application communication: filling constraints and filling validation actions.

Programmers can use a programmatic interface or an interactive interface (or both combined) in order to build and display forms. The interactive form creation is carried out with the structured graphics editor as far as the skeleton is concerned, while the FORMS system itself is used to update parts features, particularly the attached editors and procedures, create or load forms etc...

Programmers can easily integrate new editors into the system by following a few function naming constraints. They don't need any other knowledge of the system.

Generally speaking, their main task will be to write the attached procedures defining the filling constraints and validation actions which constitute the link between the form interface and the application. These procedures can be any `Lisp` functions and can modify some variable or fixed information since access to all forms components is given in the programmatic interface.

Finally, it is possible to define several types of form filling:

- passive, i.e. it is only when the user selects a filling area that the associated editor is activated,
- ordered, i.e. among ordered filling areas, the editor of the first one is called automatically, then after leaving it and possibly triggering an attached procedure, the second editor is called etc...,
- blocking, i.e. unless the form is filled up, no other input will be allowed to the user.

The Graphical Structures Editor (GSE)

GSE [20] is a tool for graphically managing hierarchies, i.e. editing all types of hierarchical structures through a graphical tree mapping in a window.

It offers trees and non circular graphs display facilities inside a window: horizontally or vertically, normally (i.e. nodes are displayed so that the tree structure is visualized with maximum clarity) or in a compacted manner (to make best use of the window space).

Interactive manipulations of the displayed hierarchies are of two types: presentation facilities and tree updating. Since nodes and links on the screen are first layer objects (lines for links, text, bitmaps, graphical objects for nodes), a restricted version of the structured graphics editor allows to move, add and delete nodes and links. This will modify the tree structure except for moving nodes. Complementary presentation facilities are opening and closing nodes (i.e. showing or not sub-nodes), displaying sub-trees or viewports (i.e. a view of the tree which fits totally into the window whatever the size and number of nodes) in other windows. The resulting trees can also be manipulated.

Semantic features of GSE cover also presentation facilities and tree updating (i.e. applications structures updating). GSE allows to declare node types, and link types defined between node types. Node types (or object spaces) are associated to a GSE window in order to display an application structure objects and link types are associated to the window in order to show one or several types of hierarchies between the objects.

Programmers can associate as well functions to the direct manipulations modifying the displayed tree structure, i.e. adding or deleting a node, adding, deleting or moving links. If a function is not successfully evaluated, the displayed tree update is undone, thus providing feedback. Such a feedback is offered by default when attempting to draw a link between nodes which types don't belong to the definition domain of the link type.

The Active Icons Manager (AIM)

The Active Icons Manager [21] allows to establish a graphical dependency link as well as a semantic dependency link between MMI objects. An object, if it has a graphical representation, can be linked to other objects so that the messages received by its representation will be transmitted to the depending objects. This allows, for instance, to drag or close related objects by only manipulating the father one. This object can also be linked to other depending objects (of any type) so that whenever it is activated somehow, a chosen message is sent to its dependents. This can be used, for example, to show an engine turning whenever a throttle is pressed down.

3. Conclusion

In conclusion, from the management point of view, the guidelines for the second year work have been clearly defined to the COCOS partners:

- Hardware group: study and design the sequential Parlog machine,
- Language group: study and design compilers for Parlog,
- MMI group: design a general purpose model for MMI.

The following list of items gives a clear definition of the tasks which have been distributed to each member of the COCOS team:

- a) Compiler for generating 68020/ μ syc instructions,
- b) Design of the internal architecture of μ syc,
- c) Selection of the parts of the SPM/run-time support to be microcoded in μ syc,
- d) Integration of the compiler on a Parlog run-time system and development system,
- e) Study of the Le - Tool scenario,
- f) Study and Development of the MMI general purpose model.

From the MMI point of view, the MMI architecture proposed in this paper attempts to offer a general framework for designing interfaces, graphical ones in particular. Most of the work has so

far been devoted to implementing the architecture three layers: they represent so far about 8000 lines of Lisp code and detailed programmers manuals have been written. We are starting now to focus on the validation of the MMI model.

The final goal of the proposed MMI environment is not to be used directly by the application interface implementor, except for the third layer tools in well suited applications, because, like application frameworks [22], it is imposing a too complex and low level of programming, but rather provide a very powerful run time kernel for supporting UIMSs and presentation control tools for all types of users.

References

- [1] Garcia, J., Jourdan, M. and Rizk, A., An Implementation of Parlog Using High Level Tools, Fourth ESPRIT Conference 1987, September 1987.
- [2] Borron, H., Types and Type-States in Le - Tool, Fourth ESPRIT Conference 1987, September 1987.
- [3] Goals definition of the working groups, COCOS technical document T.001, September 1986.
- [4] Clark, K. and Gregory, S., Parlog: Parallel Programming in Logic, ACM transactions on Programming Languages and Systems, Vol. 8, N° 1, January 1986, pp. 1-49.
- [5] Cucinelli, B., Garcia, J. and Rizk, A., Integration of μ sync in the Metaviseur Workstation, COCOS technical document T.020, July 1987.
- [6] Borron, H., Le - Tool, COCOS One Year Deliverable, Annexe 5.
- [7] Cutcher, M., Parlog Language: Annual Report, COCOS One Year Progress Report, January 1987.
- [8] SOMIW Annual Report, December 1986.
- [9] Intelligent WorkStation, Publishable Report, March 1987.
- [10] Naffah, N., White, G. and Gibbs, S., Design Issues of an Intelligent Workstation for the Office, NCC 1986 proceedings, pp. 153-159.
- [11] Tueni, M., Li, J., Fares, P., Supporting Execution and Monitoring of Office Tasks and Administrative Procedures, IWS deliverable E/r1, Bull, March 1987.
- [12] Gettys, J., Newman R. and Scheifler, R., Xlib - C Language X Interface Protocol Version 11, February 1987.
- [13] Gettys, J. and Scheifler, R., The X Window System, July 1986, Transactions on Graphics N° 63.
- [14] News Technical Overview, SUN microsystems, March 1987.
- [15] X Toolkit: A Proposed Architecture, Digital Equipment Corp., Hewlett-Packard Co. and MIT Project Athena, March 1987.
- [16] Schmucker, K., MacApp: an Application Framework, Byte magazine, August 1986, pp. 189-193.
- [17] Simonoff, J., MacApp Programmer's Guide, Apple Technical Publications, July 1986.
- [18] Chailloux, J. and al., Le - Lisp version 15.2, le Manuel de Référence, INRIA, May 1987.
- [19] Texier, M., Système de Gestion de Formulaire pour des Applications Bureautiques, Bull MTS, Direction des Etudes Avancées, September 1985.
- [20] Jureidini, G., GSE: a Software Package for Graphical Drawing and Direct Manipulation of Structures, Bull MTS, Direction des Etudes Avancées, April 1987.
- [21] Fares, P., Représentation de Connaissances et Graphisme, Bull MTS, Direction des Etudes Avancées, June 1986.
- [22] Coutaz, J., The Construction of User Interfaces and the Object Paradigm, ECOOP'87 proceedings, June 1987, pp. 135-144.

AN IMPLEMENTATION OF PARLOG USING HIGH-LEVEL TOOLS

José GARCIA[†], Martin JOURDAN[‡] and Antoine RIZK[‡]

[†] BULL – CRG, 68 route de Versailles, BP 3, 78430 LOUVECIENNES (FRANCE), ☐ [33] (1) 39.02.58.63, telex 697 030 F.

[‡] INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (FRANCE), ☐ [33] (1) 39.63.54.35, telex 697 033 F.

This paper presents the implementation of the parallel logical programming language PARLOG undertaken in the COCOS project. This implementation is based on the SPM (Sequential PARLOG Machine) virtual machine designed by the authors of PARLOG, but two aspects make it original : the SPM instructions are directly executed by μ SyC, a microprogrammable symbolic coprocessor, and the PARLOG to SPM compiler is developed with the aid of high-level tools (an attribute grammar system and a parser generator). After a brief presentation of PARLOG and the SPM and a global view of the implementation, these two aspects are presented with more details. In particular it is shown how μ SyC is well suited for the execution of PARLOG and how high-level tools drastically reduce the compiler development effort.

1. INTRODUCTION

The COCOS project aims at defining an open architecture for a multipurpose workstation and at building a number of (hardware and software) components for this architecture. A multi-layered view of what an application is has been defined. At the top lies the man-machine interface and the application description ; at the bottom lies the implementation language and the hardware. For more details on the COCOS project see the companion paper [14].

Among the various scenarios (interpretations of this multi-layered view) explored in the project, one makes use of the parallel programming language PARLOG [3] as the implementation language. This language provides many features which make it suitable for the kind of applications targeted by COCOS and which respond to the flexible approach we promote : implicit and explicit parallelism, lightweight processes, easy implementation of object-oriented paradigms, execution through compilation rather than interpretation. Furthermore its semantics is rather clearly defined and its implementation is easy on sequential machines. These aspects are detailed in section 2, which also contains a brief introduction to PARLOG.

The authors of PARLOG at Imperial College (London, UK) have defined various execution models and various virtual machines for this language. Because of the low amount of resources allotted to COCOS we have based our implementation on one of these execution models, the AND/OR process tree [4], and one of these virtual machines, the Sequential PARLOG Machine (SPM) [9], which, as the name suggests, maps easily to sequential hardware. However, considering the insufficient performance of the IC SPM implementation (written in C), and given that one of the partners (BULL) had significant background with symbolic coprocessors, we chose one of their products named μ SyC (microprogrammable symbolic coprocessor) [5] as the hardware base for executing PARLOG. A global view of the execution chain is given in section 3, together with a brief presentation of the AND/OR tree model and the SPM. Section 4 describes with more details the emulation of SPM by μ SyC and μ SyC itself.

For various strategic reasons we also decided to rewrite our own PARLOG to SPM compiler. Since this task would have required too much manpower to be done by hand, and since one of the partners

(INRIA) is involved in high-level compilation tools, we used those tools to develop the compiler. Section 5 briefly describes attribute grammars, the tools we have used — the FNC-2 attribute grammar processing system and the SYNTAX™ scanner and parser generator — and the PARLOG compiler they have generated. It is also shown how FNC-2 and SYNTAX have drastically reduced the development costs.

2. PARLOG

Intrinsic to the resolution procedure that governs the execution of logic programs, there are two main areas of non-determinism giving the potential for concurrent interpretation : firstly, in the selection of the predicate call in the body of a clause and secondly in the selection of the unifying clause from the set of clauses constituting the corresponding predicate. The two forms of parallelism resulting from these interpretations are termed AND and OR respectively [4].

PARLOG [3] is a concurrent programming language that features AND parallelism with stream inter-process communication and OR parallelism with don't care non-determinism using guarded Horn clauses. A program in PARLOG is a finite set of clauses of the form :

$$H \leftarrow G_1, \dots, G_n : B_1, \dots, B_m.$$

The declarative semantics of this clause is as in Prolog, that is, H is true if G₁ and .. G_n and B₁ and .. B_m are true. The operational semantics is however quite different. The ':' is the *commit* operator and is the parallel counterpart of the Prolog *cut*. A clause is a candidate for selection if the arguments in the head unify and also the guard succeeds. Selection amongst candidate clauses is effected at random, hence the don't care non-determinism as opposed to the don't know non-determinism of Prolog.

As can be seen, PARLOG does not need backtracking, instead the guards must be selected sufficiently informative such that the program behaves as with intelligent backtracking. This *sufficiency* of guards means that, if a clause is selected, then either a solution can be found or no solution could be found with the other clauses. Another property that guards must possess is that of *safety*, which means that a guard must not be able to bind variables in the head of the clause ; there is then no need for multiple binding environments. Guards safety can be checked almost fully at compile time and suffices for most purposes. For a complete check, including when guards contain *metacalls*, an efficient run-time check will be required.

As well as the declarative and the procedural reading of logic programs, programs written in PARLOG exhibit a *process* interpretation. A process is a predicate call, and communication between processes is effected through the instantiation of variables common to such calls. Synchronization between processes and the direction of communication is governed by the explicit declaration of the argument *modes*, input or output, for each predicate. The declaration of modes reduces much of the unification process to compilable filtering/pattern-matching and establishes directionality in the communication channels. Synchronization of processes is achieved by suspension on input variables until they are instantiated. An example of "quicksort" in PARLOG is given below (with the definition of the "partition" predicate being left to the reader) :

```

mode quicksort (? , ^) .
mode qsort (? , ^ , ^) .
mode partition (? , ? , ^ , ^) .

quicksort (unsorted,sorted) <- qsort (unsorted, sorted, []) .
qsort ([h|unsorted, sorted, rest) <-
    partition (unsorted, h, smaller, larger),
    qsort (smaller, sorted, [h|sorted1]),
    qsort (larger, sorted1, rest) .

qsort ([], rest, rest) .

```

Another distinguishing feature of PARLOG is the provision of parallel, sequential and neutral conjunction operators which specify that relation calls or search for candidate clauses is to be effected in parallel, serially, or at the implementer's will respectively. The example below shows the specifica-

™ SYNTAX is a trademark of INRIA.

tion of the more deterministic “append” operation. In this case, search for the candidate clause is specified sequential by the operator ‘;’ as opposed to ‘|’ above.

```
mode append (?,?,^).
append ([head|a], b, [head|c]) <- append (a,b,c) ;
append ([], b, b).
```

PARLOG can be translated automatically to a modeless form called Kernel PARLOG or KP. KP is also machine independent and provides a more declarative emphasis. The idea behind KP is that head arguments are replaced with generated dummy variables with which the real arguments are then unified explicitly in the guard and the body. “Append” above becomes :

```
append (p1, p2, p3) <- [head|a] <= p1, b <= p2 :
                        p3 <= [head|c] & append (a,b,c) ;
append (p1, p2, p3) <- [] <= p1, b <= p2 : p3 <= b .
```

The one-way unification primitive ‘<=’ can also be used in full PARLOG . KP therefore forms part of the definition of PARLOG and should be accepted as PARLOG .

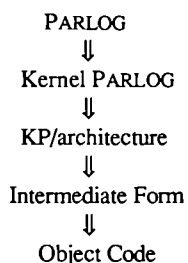
PARLOG is the latest of a family of similar languages that derives from the Relational Language [2] and includes Concurrent Prolog [17] and Guarded Horn Clauses [19]. The advantages of PARLOG over others in the same family are many : control over the granularity of parallelism, modeless option through Kernel PARLOG and most importantly guard safety property : PARLOG guards can be checked for safety at compile time which is more efficient than the run-time check of GHC and removes the need for multiple binding environments as in CP.

A major advantage of PARLOG is that it lends itself to various models of programming, such as the communicating sequential processes and the object oriented. In the first model [16], a system of communicating processes is represented by a PARLOG query and the component processes are all perpetual. A perpetual process reduces itself to itself through recursion and contains its local state in its non-shared arguments. In the second model [18, 12], objects are represented and instantiated the actor way as perpetual processes, and inheritance is performed by filtering and delegation. These two models are now being investigated in the COCOS project for the implementation of the MMI and the application layers [14].

3. GENERAL PRESENTATION OF THE PARLOG IMPLEMENTATION

3.1. Outline of the Compilation Scheme

By design, PARLOG is meant to be independent of any specific architecture, although it does not lend itself naturally to loosely coupled systems. The adaptation of PARLOG to different machines is therefore purely through the translation process. A typical such translation process and the various steps involved are illustrated below :



The first choices to be made in such an implementation are the selection of the execution model and the representation of this model in a way adaptable to the underlying intended class of machines. As was mentioned in the introduction, the preliminary aim of the PARLOG implementation in COCOS is currently to provide a vehicle for the support of the man-machine interface and the application layers, with efficiency considerations for conventional sequential machines. An easy and natural choice was then to exploit the models already developed by the PARLOG authors, namely an AND/OR tree execution model and an intermediate form based on the SPM (Sequential PARLOG Machine) [9].

PARLOG, and is used in the implementation of PARLOG on conventional single processor and shared memory multiple processor machines. As the name suggests, in the AND/OR process model there are two types of processes : the AND process, which is created to solve a conjunction of one or more literals, and the OR process, one of which is created by an AND process for each literal in the body. An OR process in turn creates an AND process for each clause in the procedure. A PARLOG computation can then be described by a tree of AND/OR processes with the initial goal statement defining an AND process at the root of the tree.

The main central data structure in the SPM is the *process tree*, which represents the active part of an AND/OR tree. Each process is represented by a *process descriptor* containing all the information needed for the execution of the process. In addition to the process tree there are the *global registers* and the *run-time queues* necessary for the suspension and scheduling of processes. Memory is divided into constant, code and heap space. The heap is further divided into equal size tagged cells for data, and equal size segments for process descriptors and argument vectors.

The main fields of the process descriptor together with their descriptions are listed below :

Parent, Sibling, Offspring : links of the process descriptor in the process tree.

Status : this can be RUNNABLE/SUSPENDED if the process is a leaf in the tree, or WAIT/NOWAIT otherwise. The latter status reflects the sequential and parallel conjunction operators in PARLOG.

wvar : reference to the variable cell on which the process is suspended.

wsib : sibling process in runnable list or suspension list.

type : process type (AND or OR).

p, a : pointers to the code and the argument vector of the process respectively.

The main global registers and flags include pointers to the heap, list of runnable processes, root process in the tree, and currently executing process. In addition the SPM maintains a *process buffer* for optimization. The process buffer holds the newly created process until it either terminates or changes its status, in which case only a process descriptor is allocated for the process. The process buffer holds the subset of the process descriptor required, namely the code and argument vector pointers. The 'ai' in the SPM code above refer to elements of the argument vector (data registers) of the current process.

Manipulation of the data structures results from the execution of the process code. The major control instructions are the ones that influence the process tree. In the "append" example above, the FORK instruction is responsible for the creation of new processes ; it specifies the type of the created process and the WAIT status. COMMIT is executed when both unification and the guards succeed. The function of the COMMIT is to prune the sibling branches of the most immediate OR parent. Suspension of processes on unbound cells by the DATA instruction transforms the cell into a pointer to a suspension queue. Once the variable represented by the cell is bound by the ASSIGN instruction, processes are added to the runnable queue. If the variable is ASSIGNed to another unbound variable, the two suspension lists are then concatenated. PAR-PROCEED effectively terminates the current process, removes it from the process tree and run-time queues and resumes the parent. Finally, term matching operations are performed by the GET-prefixed instructions and the allocation of cells/construction operations by the PUT operations.

4. IMPLEMENTATION OF THE SPM ON μ SyC

4.1. Presentation of μ SyC

μ SyC, "Micro-Programmable Symbolic Coprocessor" [5], has been primarily oriented toward relational database applications and symbolic processing. As a matter of fact both topics share a set of common requirements at low hardware level. μ SyC is seen by the main CPU as a full DMA coprocessor with a non-multiplexed 32-bit data bus and 32-bit address bus. It is designed using a HCMOS 2- μ m 2-metal layers technology and will fit into a 10 mm x 10 mm silicon area packaged in a 224-pins chip (59 for test, clock and power supply and 165 for logical signals).

The μ SyC control store is intended to be put outside the chip itself in order to easily customize its firmware without altering its internal architecture.

The internal resources available within a micro-instruction are the following :

- a 32-bit data processing unit, which performs a broad range of arithmetic and logical operations on data information ;
- a 32-bit address processing unit, which is in charge of manipulating address information ; three address spaces can be distinguished : an incoming data space, a code space and an outgoing data space ;
- a counter unit allowing a fine control of the three address space flows ;
- a tag checker allowing conditional branches to be performed according to the value of the tag ; in fact, support for both internal data types (full 32-bit data or 4-bit tag + 28-bit data) is provided ;
- a basic pattern-matching unit ;
- a sequencer which, in particular, supports multi-way branches according to the value of the tag ;
- a set of 67 32-bit registers distributed around all the units described above, plus 11 registers and counters (of 2, 6 and 8 bits) ;
- an independent I/O controller which is in charge of performing I/O operations the micro-machine asks for ; this includes the control of external handshake with all kind of I/O ports, the accommodation of retry operations and the handling of any bus anomaly. This unit operates concurrently with the micro-machine and only interferes with it when a bus error occurs or when the micro-machine requires an I/O operation while the previous one is not yet complete.

4.2. μ SyC versus a General Purpose CPU to Execute SPM

Considering that the three major features of the SPM computational model are frequent accesses to memory, frequent process switching and repetitive tag checking operations, the advantages of using μ SyC to execute SPM over a general purpose CPU are the following :

- The whole SPM instruction set is directly microprogrammed and included in μ SyC's firmware, and its execution can therefore be optimized. The SPM instruction set becomes in fact the μ SyC instruction set. Scheduling and garbage collection tasks are also carried out by μ SyC.
- Access to memory is optimized thanks to the high degree of parallelism within the micro-machine and between the micro-machine and the I/O controller (I/O can be anticipated).
- Its set of 67 internal registers allows to internally hold the whole set of variables manipulated by the SPM machine and even to buffer at least one process descriptor. A multiple register window approach (to keep several process descriptors) would not bring a significant performance improvement as SPM can fork several thousand processes during the execution of a single program.
- its tag checking capabilities are probably the best feature of μ SyC chip for executing SPM code, knowing that this is one of the most frequently executed operations (μ SyC supports up to 16 data types).

4.3. Benchmarking the μ SyC Chip and the 68020

In order to make a comparison between μ SyC and an existing CPU (the 68020) we have chosen the SPM "append" code given earlier. To perform this benchmark we made the following assumptions :

- μ SyC 's micro-cycle lasts 100 ns ;
- μ SyC I/Os are performed without any wait state within 2 micro-cycles ;
- the data structure manipulated is 4-bit tag + 28-bit data ;
- the 68020 uses a 16 MHz clock and does not require any wait state accommodation ;
- we will finally assume, for the 68020, that all its instructions are located in its internal cache.

Assuming that the length of the first list to be appended is n , the estimated execution times for the SPM "append" are respectively (both times are given in ns) :

$$t_{\mu\text{SyC}} = 15500 n - 700$$

$$t_{68020} = 68240 n + 5980$$

For e.g. a list of 10 elements, $t_{\mu\text{SyC}} = 154300$ ns and $t_{68020} = 688380$ ns, which means that μ SyC is 4.5 times faster than the 68020 (with favorable assumptions for the 68020). Furthermore, a later version of μ SyC will implement an instruction cache which will allow a significant speedup thanks to high SPM code compactness.

5. IMPLEMENTATION OF THE PARLOG COMPILER

5.1. Attribute Grammars

Attribute grammars (AGs) have been devised by D.E. Knuth [13] to describe syntax-directed computations such as those encountered in compilers and translators. They are an extension of context-free grammars in which each non-terminal is associated with a set of *attributes*: these are names for single-assignment variables which will be attached to the corresponding nodes of derivation trees. These attributes are divided into *inherited* ones, which propagate information from the root of the tree down to the leaves (context), and *synthesized* ones, which propagate information from the leaves up to the root (results). With each production comes a number of *semantic rules* which define how output attributes of the production are computed in terms of input attributes; output attributes are synthesized attributes of the LHS non-terminal and inherited attributes of the RHS non-terminals, while input attributes are synthesized attributes of the RHS non-terminals and inherited attributes of the LHS non-terminal.

Given a derivation tree of the underlying context-free grammar, an *attribute evaluator* is in charge of computing all the attribute instances attached to the nodes of the tree, using the corresponding semantic rules. The only constraint governing this evaluation is that no attribute instance may be computed before all the attribute instances used in the semantic rule defining it are evaluated. The result of this process is either the whole decorated tree or (a subset of) the attributes of the root.

When used for compiler specification, attributes represent semantic information attached to structures of the source language, such as the type of an expression, the mode of an identifier, the symbol table, and the generated code, and the semantic rules express e.g. the typing and scope rules and code generation. A typical, but simplified, example is presented in Figure 1, with (hopefully) obvious notations. However, AGs are not restricted to compiler specification. Indeed they can describe any syntax-directed computation; see part III of [6] for further references.

```

<expr> = <expr> "+" <term> ;
  $env(<expr>.2) := $env(<expr>.1) ;
  $env(<term>) := $env(<expr>.1) ;
  $type(<expr>.1) := if $type(<expr>.2) = integer and
                    $type(<term>) = integer then
                      integer
                    else real ;
  $code(<expr>.1) := { code for a stack machine }
                    concat ($code(<expr>.2), $code(<term>),
                            if $type(<expr>.2) = integer and
                              $type(<term>) = integer then
                                "add integer"
                              else "add_real") ;

<variable> = %identifier ;
  $type(<variable>) :=
    let denotation :=
      lookup ($text(%identifier), $env(<variable>)) in
    if undeclared (denotation) then
      error "Undeclared identifier" value no-type
    elseif not is-a-variable (denotation) then
      error "Not a variable" value no-type
    else extract-type (denotation)';
  $code(<variable>) := ... ;

```

Figure 1. Typical example of part of an AG.

It can be seen from this brief presentation that AGs are an attractive specification tool because of the following qualities:

- an AG is a *declarative* specification: it is not defined how the attributes are computed, but only what values they should have in the end;
- it is a *structured* specification: the underlying grammar is a natural way to separate and relate the

- various components of an AG, namely the semantic rules ;
- an AG is *locally-scoped* : each production is a “black box” which is independent from the others, and there is no global variable ;
- there is no side-effect in an AG : each semantic rule is purely *applicative* ;
- furthermore an AG is an *executable* specification : the above defined attribute evaluator can be automatically constructed.

To demonstrate the productivity gain achievable using AGs, just notice that one of the authors has written and completely tested a full ISO-Pascal to P-code compiler in just three men-months [10].

The probably most important advantage of AGs is that these specifications are executable : it is possible to construct automatically the attribute evaluator corresponding to a given AG. However this is not a trivial task because of two main problems : first, the evaluation is a priori non-deterministic, and the order in which the tree is traversed and the attributes are evaluated is not obvious ; secondly the number of attribute instances to evaluate and store grows rapidly with the size of the input text, and storage consumption problems arise. A large body of research work (see main results and bibliography in [6]) has been devoted during the last twenty years to the construction of efficient evaluation methods. These researches are now mature enough to construct evaluators which can compete with hand-written programs. A number of systems (more than 30 are reviewed in [6]) have been developed to implement an evaluation method. Some of them can be considered as production-quality, e.g. GAG and LINGUIST, and have actually been used in industrial projects to build (parts of) compilers. The COCOS project uses the FNC-2 system and associated tools developed by the “Langages et Traducteurs” project at INRIA.

5.2. The FNC-2 and SYNTAX Systems

FNC-2 is a new, production-quality AGs processing system. It is *powerful* since it accepts the strongly (or absolutely) non-circular class of attribute grammars, a very large subclass including all the practical examples encountered in compilation. It is *efficient* since it is now possible to transform a strongly non-circular AG into an L-ordered AG — to which corresponds a completely deterministic evaluator based on visit sequences — of nearly the same size. Another advantage of those evaluators is that they allow to use storage management optimization techniques. The net result is thus the production of fast and memory saving evaluators.

The *ease of use* of FNC-2, measured as the effort necessary to write and debug an AG, stems from the conjunction of two concepts :

- the syntactic base of FNC-2 AGs is an abstract syntax, allowing the author to get rid of sometimes cumbersome syntactic constraints ;
- the AG description language OLGA is a specialized language designed for this unique application, and not an existing language simply augmented with notations for accessing attributes. OLGA is an applicative and strongly typed language, two conditions necessary to provide the AG writer with absolute programming safety. In addition OLGA includes other modern concepts such as polymorphism, exception handling and (system-managed) modularity.

The *versatility* of FNC-2 is due to the total independence of the specification w.r.t. its implementation(s), which implies that, starting from the same AG, several evaluators can be generated, differing in the implementation language (C, Lisp and Ada™), in their evaluation mode (exhaustive or incremental), and in the tools they are connected to, upstream or downstream.

SYNTAX is a complete system for automatic construction of efficient scanners and parsers ; it is composed of four main parts :

- a *grammar reader* accepts a grammar written in BNF and transforms it into an internal form ;
- an LALR(1) *syntactic constructor* builds from this internal form a bottom-up stack automaton (a parser recognizing the language) ; an *optimizer* reduces by a large amount the size of this parser ; if LR conflicts are detected, they can be solved using user-supplied easy-to-write precedence rules ; furthermore the user may write actions and predicates to influence the syntactic analysis ;
- a *lexical constructor* takes as input the internal form of a grammar and the lexical description of its terminals (in form of regular expressions) and directly produces, using LR techniques, a finite-

™ Ada is a registered trademark of the Government of the USA, AJPO.

state deterministic automaton (the scanner of the language); this scanner may use an *unbounded* number of look-ahead characters to solve the conflicts and can be associated with user-defined actions and predicates;

- a *run-time kernel* will form the core of the lexico-syntactic analyzer of the language; this kernel includes a scanner and a parser which will interpret the former tables, together with many modules allowing the easy and cheap construction of an efficient and powerful analyzer: management of the identifiers table, of comments, of error messages, of error recovery, etc.

Error recovery is based on a three-level scheme (local correction, forced insertion, global recovery); this recovery is automatically built into the scanner and the parser without modifying the descriptions, and the grammar writer may also tune it with a set of easy-to-write patterns. The power and efficiency of this error recovery scheme is the most striking feature of SYNTAX.

FNC-2 and SYNTAX are being developed on UNIX; they are written in C and will eventually be "bootstrapped" in part. A prototype version of FNC-2 will run before the end of the year, while SYNTAX is ready for distribution. For more details on FNC-2 and its theoretical principles, see [11] and [15]. For more details on SYNTAX, see [1].

5.3. The PARLOG Compiler

The structure of the PARLOG compiler is shown in Figure 2. `parlog.lecl` contains the lexical specification of PARLOG in terms of regular expressions and simple rules. The possibility of declaring synonyms very easily in *lecl* makes the substitution of PARLOG neutral operators more simply performed at lexical analysis time. For the SPM implementation these are automatically replaced by the sequential counterparts. `parlog.recor` contains specification for SYNTAX error recovery. The power of error recovery and correction produced by SYNTAX are significant. Bearing in mind that in logic programming almost all syntactic errors are of the missed bracket and comma type, a syntactically erroneous PARLOG program is fully recovered.

The main parts of the compiler are `parlog.atc`, `par-kp.olga` and `par-spm.olga`. `Parlog.atc` consists of the PARLOG concrete grammar expressed in BNF, together with tree construction operations attached to each production rule. It consists of 56 production rules and 250 lines of *atc* code. The output of this phase is the representation of the PARLOG source program in abstract tree format; nested parallel constructs are also flattened into same level parallelism. The output abstract trees are specified by the abstract grammar in `parlog.asx` and constructed in *atc* directly from the operators of the grammar; this consists of 8 phyla and 21 operators. `Parlog.prio` contains disambiguation rules to be used by the *lalr1* parser generator.

`Par-check.olga` is the attribute grammar, written in OLGA, which checks the static semantics of a PARLOG program. As with all logic programming, this part is minimal and consists mainly of checking the existence of a called predicate and of the construction of a symbol table which records each predicate with the modes of its arguments, to be used in subsequent phases. In this phase also, optimizing source transformations on a PARLOG program can be done, as well as complicated compile-time checks such as guard safety. The size of this phase with the symbol-table module is 300 lines, of which 70 are for the symbol table. The latter is implemented as an abstract data type, represented as a simple list with access functions, and constructed as a compound attribute attached to the Predicate phylum of PARLOG and finally used as a global exported attribute.

`Par-kp.olga` performs the translation from PARLOG trees directly into KP/and-or-tree and `par-spm.olga` translates the latter into SPM code. The size of the two phases together is about 3000 lines of OLGA code without taking into account default OLGA semantic rules, automatic generation of attributes, etc... The main interface between these two phases is `kpao.asx` which contains the specification of the abstract grammar of KPao. Very similar to PARLOG syntax with the omission of modes and the addition of machine primitives, this syntax consists of 8 phyla and 27 operators plus operators for certain additional built-in predicates. The main attributes in `par-kp.olga` are one synthesized for the construction of the output term, and two compound that hold the renaming of variables and their occurrences respectively.

The advantages of OLGA in the production of the compiler have been significant. Firstly, the entire front end necessitated a period of approximately three month (for one untrained man) without taking into account familiarity with OLGA and SYNTAX. Secondly, the decomposition of the compiler

into phases is a natural and simple process, the interface between phases being described by an abstract grammar in OLGA. Thirdly, there is a close correspondence between functional attribute grammars and logic programs [7] such that the OLGA attribute grammars were derived nearly mechanically from the logical specifications found in several papers from Imperial College.

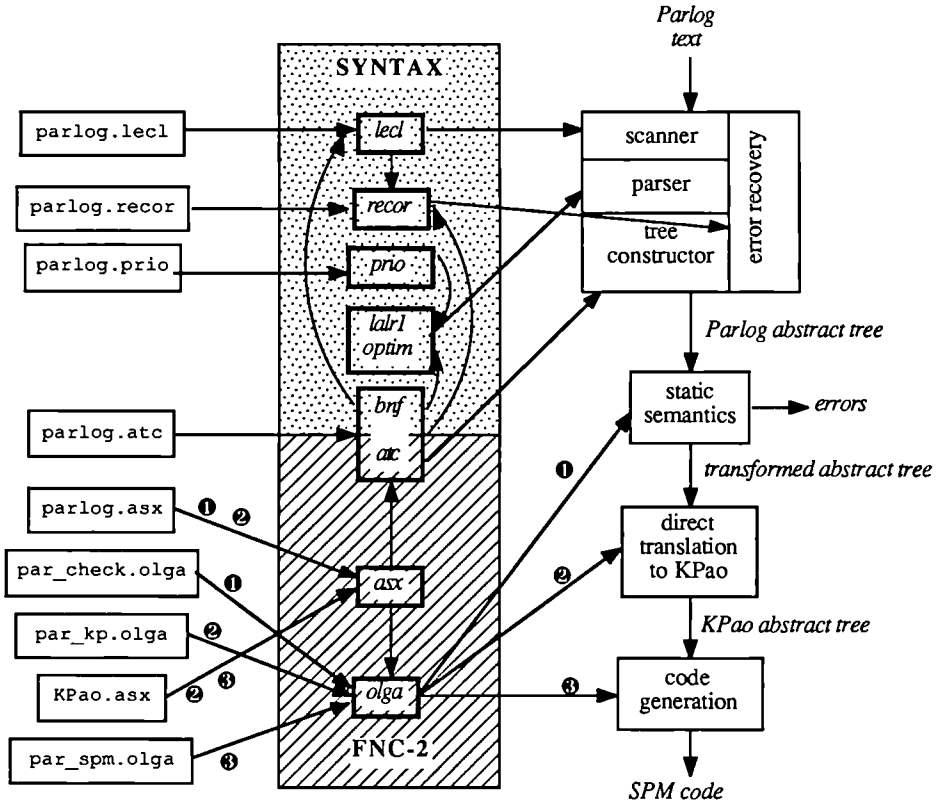


Figure 2. Structure of the PARLOG compiler.

6. CONCLUSION

This paper presented the implementation of PARLOG undertaken in the COCOS project. The two original aspects of this work are the use of a symbolic coprocessor as the hardware execution base and the use of an attribute grammar system and a parser generator for the development of the compiler. The coprocessor allows reasonable execution efficiency of this parallel language on a sequential machine, while the high-level tools reduced the compiler development costs by a very important ratio. Both these aspects are hence an advance in information processing technology, thereby responding to the aims of ESPRIT.

REFERENCES

- [1] P. Boullier, P. Deschamp & B. Lorho : *The SYNTAX Reference Manual*, to appear, INRIA, Rocquencourt (1987).

- [2] K. Clark & S. Gregory : "A Relational Language for Parallel Programming", *ACM Conf. on Functional Programming Languages and Computer Architecture*, Portsmouth, pp. 171-178 (1981).
- [3] K. Clark & S. Gregory : "PARLOG : Parallel Programming in Logic", *ACM Trans. on Progr. Languages and Systems* 8 (1), pp. 1-49 (1986).
- [4] J.S. Conery : *The AND/OR Process Model for Parallel Interpretation of Logic Programs*, PhD thesis, University of California, Irvine (1983).
- [5] M. Couprie, J. Garcia, T. Maréchal & D. Terral : *μSyC : Coprocesseur Microprogrammable pour les Applications Symboliques*, rapport DSG/CRG/87020, BULL – CRG, Louveciennes (1987).
- [6] P. Deransart, M. Jourdan & B. Lorho : *A Survey on Attribute Grammars — Part I : Main Results, Part II : Review of Existing Systems, Part III : Classified Bibliography*, rapports RR-485, RR-510 et RR-417, INRIA, Rocquencourt (1985-1986). A revised version is about to be published by Springer-Verlag (LNCS series).
- [7] P. Deransart & J. Maluszynski : "Relating Logic Programs and Attribute Grammars", *J. of Logic Programming* 2 (2), pp. 119-155 (1985).
- [8] I. Foster : *The Compilation of PARLOG for the Sequential PARLOG Machine*, research report, Dept. of Computing, Imperial College, London (1985).
- [9] S. Gregory : *The Sequential PARLOG Machine*, research report, Dept. of Computing, Imperial College, London (1985).
- [10] M. Jourdan : *Les Grammaires Attribuées : Implantation, Applications, Optimisations*, thèse DDI, Université de Paris VII (1984).
- [11] M. Jourdan & D. Parigot : *The FNC-2 System User's Guide and Reference Manual*, draft, INRIA, Rocquencourt (1987).
- [12] K. Kahn, E.D. Tribble, M.S. Miller & D.G. Bobrow : "Objects in Concurrent Programming Languages", *OOPSLA'86*, Portland, pp. 242-257 (1986).
- [13] D.E. Knuth : "Semantics of Context-free Languages", *Math. Systems Theory* 2 (2), pp. 127-145 (1968).
- [14] N. Naffah & M. Texier : "COCOS Architecture and its MMI Model", in this volume.
- [15] D. Parigot : *Mise en Œuvre des Grammaires Attribuées : Transformation, Evaluation Incrementale, Optimisations*, thèse de 3^{ème} cycle, Université de Paris-Sud, Orsay (1987).
- [16] G. Ringwood : *The Dining Logicians*, research report, Dept. of Computing, Imperial College, London (1985).
- [17] E.Y Shapiro : *A Subset of Concurrent Prolog and its Interpreter*, tech. report TR-003, ICOT, Tokyo (1983).
- [18] E.Y Shapiro & A. Takeuchi : "Object Oriented Programming in Concurrent Prolog", *New Generation Computing* 1 (1), pp. 25-48 (1983).
- [19] K. Ueda : *Guarded Horn Clauses*, tech. report TR-103, ICOT, Tokyo (1985).

TYPES and TYPE-STATES in LE_TOOL

Henry BORRON - INRIA Sophia Antipolis - O6560 VALBONNE - FRANCE *

LE_TOOL PROJECT

The final aim of LE_TOOL project is a portable programming environment (with a sophisticated syntax-directed editor and an important extensible software library). Its intermediate goal is a general-purpose Typed Object-Oriented Language.**

More precisely, productivity and reliability are the two major points aimed at by LE_TOOL. Efficiency, flexibility, user-friendliness (especially, readability & syntax-directed editing) come just afterwards.

Roughly speaking, LE_TOOL inherits productivity from object-orientation, while reliability and efficiency come from types and type-states. As such, major influences are SMALLTALK[1,2,3], CLU[4,5,6] and IBM's NIL[7,8].

A first implementation of the language is scheduled for March '88 (in Le_Lisp [21], on a SUN 3.75 workstation).

SUMMARY

An *overview of the language* is first given together with simple examples. It introduces the tree of types and the tree of "partitions" (classes).

The semantics of types is then discussed; the central role of type "interfaces" is stressed. Next, are explained the basic algorithms for *computing interfaces* a) of a parameterized type (given the interface of the associated type generator and the interfaces of the type parameters), b) of a type knowing its partitions.

Finally, the adaptation of *type-state checking* to LE_TOOL is shown to be simple, and its use not too demanding a constraint to the programmer.

The conclusion evokes briefly why features preceedently discussed make *syntax-directed editing* a very natural and powerful tool of the envisaged system.

* This one-man project started in '83-'84 at INRIA (Sophia-Antipolis - FRANCE). Since June '86, it is partly supported by "COCOS" (ESPRIT project # 956) thru a subcontract from BULL-MTS/DEA (Massy-Palaiseau - FRANCE).

** Hence, the name of the project.

1. LE TOOL LANGUAGE : A SHORT PRESENTATION

1.1 LE TOOL Is an object-oriented language

- In other words, LE_TOOL implements five methodological rules :
 - center modules around data-structures ;
 - make data-structures simples (linears) ;
 - prohibit direct access to data-structures (use procedures) ;
 - make data-structures ("objects") responsible (pointing to their dictionary of "methods") ;
 - factorize similarities between types of objects (eg. : "inheritance")

Major design decisions [17] concern points that would have made LE_TOOL more uniform, and that were consciously abandoned in favor of more practical purposes. LE_TOOL is not an actor language [9, 10, 11]. It has no metaclasses[1, 12]. It does not support multiple inheritance [13, 14, 15, 16]. Blocks of code are not objects [1].

- LE_TOOL supports a *tree* of types. A type has an *interface* listing what messages can be "sent" to the type itself, and sent to its instances (Ex. : the interface of Circle).

A type may have parameters, in which case it is termed a *type generator*. *Parameters* may be simple *constants* (like an integer). Parameters may be formal *types* too. A formal type may be constrained or not. A constrained formal type is attached an interface describing what methods a candidate type should support to be a valid actual parameter.

- The implementation of a type is "cut into pieces", termed "*partitions*" (classes). Partitions form a *tree* that parallels the tree of types (the latter derives from the former - cf. §2.4). In addition to the SUPER field (for inheritance purpose), a partition has SLOTS and METHODS.
 - SLOTS are further distinguished into TYPE and INSTANCE slots, and INSTANCE SLOTS into SHARED and SPECIFIC ones*. SLOTS present a uniform view of referenced objects : no matter its category, a slot may reference either a (data-)object or a *method*. Initialization is supported for data slots.

An instance can access the specific slots of another instance only thru messages. In the same way, direct access to type slots is denied to instances (and vice-versa).

In practice, the three categories of SLOTS reflect different needs : TYPE SLOTS are mainly used for storing type-methods and facts about the group of instances; SHARED INSTANCE SLOTS for storing instance methods.

```

Ex.: type-slots =>
    "new" (in Circle)
    "read_from:", "from:to:by:" (in Integer)
    "pi" (object), "pi" (method) (in Real)
  shared instance slots =>
    "pi" (object) (in Circle)
    "abs", "+", "mod:" (in Integer)
    "radius", "radius:" (methods) (in Circle)
  specific instance slots =>
    "radius", "center" (in Circle)

```

* Note the difference with the SMALLTALK class structure (cf. class/instance variables).

- Under METHODS are listed *statically* bound methods, further partitioned into TYPE and INSTANCE METHODS (a method may be bound *dynamically* to a slot if passed as an argument in a type or an instance message).

A method describes the computation done once a message has been received. A message is composed of a selector (unary : "abs"; binary : "+"; n-ary : "from:to:by") and of zero, one or several *arguments*. The computation may return one *result* or none. A method may produce *exceptions* under certain circumstances (ex: "divide-by-zero"). Besides ordinary methods (ex. : "+"), LE_TOOL supports *iterators* that yield elements of collections in sequence (ex. : "from:to:by:").

Examples of method definitions :

```
+          # (Integer => Integer) SIGNALS (overflow)      % In Integer
from:to:by: # (Integer, Integer, Integer =>* Integer)    % In Integer
insert:at:  # (String, Integer) SIGNALS (out_of_bounds) % In String
elements   # ( =>* T)                                    % In Array[T]
```

In practice, a method lists the names of its argument variables, declares temporary variables and has a body. In LE_TOOL, this body is bracketed by *states pre- and post-conditions* (for type-state checking - Cf. §3) and made of *imperative statements* that control the flow of actions.

List of statements :

```
test          : IF*
loop          : LOOP, WHILE, REPEAT, FOR ** + BREAK, CONTINUE
assignment   : :=
compound statement : BEGIN
empty statement : VOID
cascade of calls : <receiver> <message1>, <message2>, <message3>, ...
return statements : RETURN, YIELD
exception management : SIGNAL, RAISE, EXCEPT, RESIGNAL
```

Actions are access/modification of the receiver's slots, and message sending (either to the receiver [termed "self" or "supra" ****] or to an other object). In a given expression, message sending is done according to *precedence rules* (unary messages have the highest precedence ; n-ary the lowest ; binary are ranked into several categories reflecting usual habits *****).

1.2 LE_TOOL is a typed language

- ° LE_TOOL is typed in the sense that declarations of slots, variables and parameters mention the types of objects that are going to be plugged in.

```
Ex.: data-slots =>
      *pi   # Real := Real pi           (in Circle)
      radius # Real                    (in Circle)
      a     # Array[T]                 (in P_Queue)
```

-
- * No TYPE_CASE statement as in [19] since object-orientation makes it precisely useless.
 - ** The FOR statement allows the simultaneous and controlled use of several iterators [20].
 - *** "supra" is used for activating a shadowed method (same role as "super" in SMALLTALK).
 - **** 1+2*3 yields 7, and not 9 as in SMALLTALK.

```

method_slots =>
  abs # ( => Integer)                (in Integer)
  +   # (Integer => Integer) SIGNALS (overflow) (in Integer)
  mod: # (Integer => Integer)         (in Integer )

```

- Type declarations allow the compiler to *detect (some) inconsistencies*, and, when compared with an untyped language such as SMALLTALK, to produce a *more efficient* code by doing (at least some) method searching at compile-time instead of run-time. The run-time overhead is reduced to zero or to a simple indirection thru an index table prepared at compile-time. In a rare number of cases (ex. : x # Object), the number of entries in such a table may overgrow leading to an inflation in memory needs (table + methods). Two strategies are possible : a) run-time searching ; b) global data-flow analysis to decrease the size of the table (restricting selectors and types according to the application). In the future LE_TOOL programming environment, run-time searching will be preferred for the development phase while optimization will be done for completed applications.

An other (commercial) advantage of typing is the possibility to *isolate the useful pieces of code* out of the whole library (under the same conditions as above).

2. TYPES IN LE TOOL

2.1 Semantics of type declarations

- Declaring a slot or a variable to be of type T (x # T) means objects that are going to be plugged into the considered slot or variable will support all the methods listed in the interface. In other words, methods of a candidate interface must be compatible with those of the contract interface.
- In case T pertains to the tree of types, then "x # T" means any object either of type T or of a subtype of T can be plugged into "x".

If T happens to be an instantiable internal node of the tree of types, restriction to objects of type T only is obtained by creating a "transparent" subtype of T, say T', and using it in the declaration in place of T (x # T'). A transparent subtype is easily obtained by creating a subpartition T' that simply inherits from the partition associated to T.

2.2 Compatibility of methods

A candidate method is compatible with a contract method having the same selector if a certain number of obvious rules are respected (both methods are iterators or ordinary methods, both have the same number of arguments, both return a result or not), and if the types of arguments/result of the candidate interface are respectively compatible with the types of arguments/result of the contract interface.

The type of an argument of a candidate method is compatible with the corresponding type in the contract method if the interface of the latter is compatible with the interface of the former. In case both types are actual, this means the type of the argument of the candidate method should be equal to the contract type or to one of its *supertypes*.

The type of the result of a candidate method is compatible with the type of the result in the contract method if the interface of the former is compatible with the interface of the latter. In case both types are actual, this means the type of the result of the candidate method should be equal to the contract type or to one of its *subtypes*.

```

INTERFACE Circle
METHODS
TYPE
  new # ( => Circle)
INSTANCE
  center: # ( Point )
  radius: # ( Real )
  surface # ( => Real )
  draw # ( )
  radius # ( => Real )
  ...
END

```

INTERFACES of a SIMPLE TYPE (Circle)
and of a [constrained] TYPE GENERATOR (Set)

```

GIVEN
INTERFACE T
METHODS
  INSTANCE
    = # ( T => Boolean)
  END

INTERFACE Set[T # Type]
METHODS
TYPE
  create # ( => Self)
INSTANCE
  insert: # ( T )
  delete: # ( T )
  has: # ( T => Boolean)
  size # ( => Integer)
  elements # ( =>* T)
  = # ( Self => Boolean)
  copy # ( => Self)
  IF INSTANCE T HAS
    copy # ( => Self)
  END
  ...
END

```

IMPLEMENTATION of a PARTITION (Circle)

```

PARTITION Circle
SUPER ...
SLOTS
TYPE
  new # ( => Circle)
  ...
INSTANCE
SPECIFIC
  center # Point
  radius # Real
SHARED
  *pi # Real := Real pi
  center: # ( Point )
  radius: # ( Real )
  surface # ( => Real )
  draw # ( )
  radius # ( => Real )
  ...
METHODS
TYPE
  WHEN new DO
  BEGIN << >>
  RETURN ...
  END << >> => << >>
  ...

  INSTANCE
  WHEN center: c DO
  BEGIN << >>
  center := c
  END << center >>

  WHEN radius: r DO
  BEGIN << >>
  radius := r
  END << radius >>

  WHEN surface DO
  BEGIN << radius >>
  RETURN(pi*radius ^2)
  END << radius >>

  WHEN draw DO
  BEGIN << center, radius >>
  ...
  END << center, radius >>

  WHEN radius DO
  BEGIN << radius >>
  RETURN(radius )
  END << radius >>
  ...
END

```

2.3 Construction of the interface of a parameterized type

2.3.1 Introduction

- A parameterized type is obtained by instantiating parameters of a type generator (ex. : Array[Integer], Set[Association[Key, Value]]).
- Parameter substitution arises in two different cases :
 - a) when a variable is declared to have a parameterized type
ex. : x # Array[Integer] ;
 - b) when a partition names its parent thru the SUPER field
(See the construction of the interface of a type given its partitions §2.4)
ex. : SUPER Set[Partition[Key, Value]]).
- The parameterized type may have a more or less complicated form depending on the number and type of its parameters.
Formally, a given type generator (say TG[...]) has K formal parameters (say, F_1, F_2, \dots, F_K).
A substitution consists in giving each F_k an actual value which may be :
 - a) a simple type (ex: Character, Magnitude...);
 - c) a formal type (whose interface belongs to the context of the enclosing partition) ;
 - b) a parameterized type which may have conditional methods or not. These two cases will be respectively qualified of "partially bound" and "completely bound".

At the difference of a type generator a partially bound parameterized type has no GIVEN clause of its own : constraints such a clause would express are part of the environment of the parameterized type.

2.3.2 Overall algorithm

- When the actual parameter is a parameterized type (partially or completely bound), the compatibility problem obviously arises at the level of that parameter too, and recursively for the parameters of that parameter (if any) ...

An adequate representation is a *tree* where the leaves are either simple or formal types , and where the internal nodes are parameterized types referencing type-generators. The compatibility algorithm for substitution should be *recursive (post-order traversal)* .

- When asked to compute its interface, a parameterized type first ask all its parameters to compute their own interfaces. If one parameter fails, the parameterized type fails too. If all parameters have an interface, then the parameterized type consults its generator to know if its actual parameters fulfil the constraints listed in the type generator. If not, then the parameterized type returns false (it fails). Otherwise, it compute its actual interface before returning true (success).

2.3.3 Compatibility of actual parameters vs. the type generator

◦ Introduction

The constraints listed in the type generator (GIVEN and HAS clauses) must be satisfied. This will be done by examining if each required method has a compatible counterpart in the interfaces of candidate parameters.

The whole process is better understood by considering the whole text of the type generator interface is first copied in place of the parameterized type interface, then transformed by replacing everywhere the formal parameters by the actual ones (error detected if not in the same number) . This yields the transformed methods (cf. types of arguments/result), and also the transformed GIVEN and HAS clauses.

Finally, methods of the transformed GIVEN and HAS clauses that are satisfied are withdrawn. To be valid, the transformation must empty the GIVEN clauses.

◦ GIVEN clauses

Each method "M" of each transformed clause (one per parameter) must be satisfied.

To be satisfied, a method "m" -compatible with "M"- should be found in the candidate interface of the considered parameter. In addition, "m" should not be conditional (that is further constrained by some HAS clauses on one or several formal parameters).

◦ HAS clauses

If the methods in the GIVEN clauses are all satisfied, then the considered substitution of parameters is valid.

As a consequence, all methods of the parameterized type that are not conditioned by some HAS clauses are certainly available

Other methods (methods that were conditional in the type generator) may be available or not, and, if available, with or without further constraints. This depends on what becomes the transformed HAS clauses that are attached to each one :

- if one HAS clause aborts, then the method is not available ;
- if each HAS clause is emptied, then the method is available without further conditions ;
- otherwise, it is available only if adequate actual parameters are substituted for the formal parameters of the enclosing partition /type.

2.3.4 Algorithm

◦ Each clause is examined according to the type of its actual parameter :

- *simple type*

Methods appearing in the clause are searched in the actual interface of the parameter.

Three cases may occur :

- . absent --> the search in the clause is aborted;
- . incompatible --> the search in the clause is aborted;
- . compatible --> the method is removed from the clause.

- *formal type*

Methods appearing in the clause are searched in the actual interface of the parameter (the interface of this formal type is part of the context of the enclosing partition/type (in a GIVEN clause too!)). Three cases :

- . absent --> the method is left in the clause ;
- . incompatible --> the search in the clause is aborted ;
- . compatible --> the method is removed from the clause.

- *parameterized type*

Totally bound parameterized types behaves like a simple type since they do not have HAS clauses. So, this part concerns *partially bound* parameterized types.

Methods appearing in the clause are searched in the actual interface of the parameter. Three cases :

- . absent --> the search in the clause is aborted ;
- . incompatible --> the search in the clause is aborted ;
- . compatible --> the method is removed from the clause,
and, if the method was conditioned by HAS clauses, then these HAS clauses are added to GIVEN or HAS clauses of the considered parameterized type (possible distribution over several formal types).

- A failure in the above step implies either the parameterized type to be declared invalid (if it occurs in a GIVEN clause), or a conditional method to be made unavailable (if it occurs in a HAS clause).

- Suppose all the clauses were scanned without failure. Two cases may arise depending on whether one or several methods are still present in a clause :

- present :

- . if it was a GIVEN clause, then it aborts and, so, the parameterized type is declared invalid;
- . otherwise (HAS clauses), the parameterized type has one more method available, but conditionally (and, if the method is not withdrawn by some other HAS clauses).

- absent :

- . if it was a GIVEN clause, then it is satisfied, and the parameterized type may be declared valid (if all other GIVEN clauses are valid too);
- . otherwise (HAS clauses), the parameterized type has one more method available without further conditions (but, if the method is not withdrawn by some other HAS clauses).

2.4 Construction of the interface of a type from its partitions

2.4.1 Introduction

The interface of a type T is constructed along with its implementation. The implementation of T is computed in a recursive way using partition T and the implementation of the parent of T (from which inherited methods are selected).

2.4.2 Computing the parent of T

- The parent of T is named under the SUPER field. If it is a parameterized type, the corresponding type generator is first computed, then actual parameters are substituted in place of formal ones.

Ex. : Partition generator Dictionary[K,V] inherits from partition Set[T]. Its SUPER field indicates the transformation to apply to parameter T :

SUPER Set[**Association[K,V]**]

Wherever T appears (argument or result types, clauses) it is going to be to be changed to Association[K,V]. An example is the specification of the "has:" method in Set[T] :

```
has: # ( T => Boolean)           {Set [T]}
has: # (Association[K,V] => Boolean) {Dictionary[K,V]}
```

- In the process of computing the parameterized type, transformed GIVEN and HAS clauses are to be checked (cf. §2.3.3 and 2.3.4).

In fact, it may happen that a GIVEN clause is not valid. Before concluding to an error, one must check that the considered condition is not due to one or several methods that are in fact shadowed by partition T (that is, redefined in T)

Ex. : If we try to compute the interface of Dictionary, we get in trouble since Association[K,V] should have an instance method for "=", which is not the case ... The type is valid yet. This is because partition Dictionary re-implements methods that were using the message "=" in Set[T].

- At this point, specifications of methods in partition T are checked to be compatible with specifications in the parent implementation (when redefined).

2.4.3 Deriving the interface of T

- Hence, the construction of the interface of type T
 - if the partition is constrained, its interface too and by the same *GIVEN* clause than the partition ;

- specifications of type and instance methods listed in the *definition* of the partition T are copied to the interface (with HAS clauses if any) ;
- specifications of type and instance methods listed in the *interface* of the (transformed) parent implementation are copied if not shadowed (possibly with HAS clauses) ;

2.4.4 Notes

- In case there is no problem with the GIVEN clauses of the parent of T, the interface of type T can be computed using only the definition of partitions it is made of, or say in a recursive way, the *definition* of partition T and the *interface* of the parent of T.
- As mentioned above (cf. §2.4.2), specifications of methods in subpartitions are checked for compatibility against previous specifications of the same methods (if any) in superpartitions. To add some flexibility to compatibility rules for arguments types, LE_TOOL has been augmented with a construction [18] that transfers the responsibility for compatibility *from the method level to the partition level* (possibility of implementing *one* specification thru *several* methods).

Ex. :

	{in Number}	
	+ # (*Number => Number)	
	/	\
	{in Integer}	{in Real}
+ # (Integer => Integer)		+ # (Integer => Real)
+ # (Real => Real)		+ # (Real => Real)

3. TYPE-STATES IN LE_TOOL

3.1 Introduction

- Type-state checking [7,8] permits to discover subtle and frustrating initialization errors (compromising both reliability and productivity) that wouldn't be found by standard type checking. For example, if "i" and "j" are declared as integers, assigning "j" to "i" is correct from the point of view of types, but provokes an error at run-time ... if "j" has no value. Type-state checking would detect the inconsistency at compile-time.
- Restricted to mono-processing, the *main idea* is to add *before and after the body* of each method a declaration about the states of variables in the context of activation. Checking the body of a method consists in proving the post-condition starting from the pre-condition and applying rules about calls (checking the pre-conditions and continuing with the post-conditions) and statements (simple rules apply for loop and tests statements based on invariance considerations).

3.2 Adaptation to an object-oriented language

- In the context of an object-oriented language, *only the state of the receiver is available*. (in the partition defining the considered method). The adaptation of type-state checking should take this into account.

◦ Rule 1

The **receiver's state** is *specified* on entry and exit of each method in a partition. This is done by naming the initialized slots. If the language supports exceptions management, several states might be possibly specified on exit of a method (one for the normal outcome and one for each exceptional outcomes).

◦ Rule 2

In LE_TOOL, all **arguments** in a message pattern should be *completely initialized* (when entering and exiting the method).

This should not be perceived as too constraining : if part of an object (argument) is not initialized, it is always possible to send a preliminary message to that object so that only the useful initialized part is to be transmitted. In addition, this does not fix the way arguments are to be passed. It does not mean either that arguments can't be changed.

◦ Rule 3

If a **result** is to be returned, *three cases* are to be distinguished :

- if the result is a *newly created instance* (the method being a type instance-creation method), then -exceptionnaly- the state of the object is to be described (after the state of the receiver) ;
- if the result happens to be the *message receiver* *, then its state is specified and can be retrieved ;
- otherwise, the rule is that the result must be *fully initialized* (on exit) **.

- A few other practical rules have been devised for simplifying the task of the programmer in usual situations. They are not listed here.

4. CONCLUSION

- Object-orientation meshes well with types and types-states : explicit references between modules (cf. IMPORT/EXPORT clauses in more conventional languages) disappear ; declarations of states get simpler. Related to inheritance, the tree structure of types brings flexibility to type declarations (smooth variation from strong typing to no typing at all). Conversely, types and type-states improve efficiency and reliability (early error discovery) of object-oriented systems.
- Benefits also appear at the programming environment level making *syntax-directed editing* a very natural tool in the envisaged system : automatic contextual display of possible messages to a given receiver is made possible because of the automatic resolution of inter-modules references, and because of type-state checking ; existence of type interfaces makes it natural. Automatic contextual display of (imperative) statements is nothing new but add to the user-friendliness of the system.

* A syntactic annotation in the method declaration inform the compiler about this case.

** This rule permits to compute consequences of a call (cf. 3.1.b) even when recursively involved bodies are not fully stated. Otherwise, this computation would not be so simple !]

ACKNOWLEDGEMENTS

This project would not have been launched without the confidence of Gilles KHAN and Pierre BERNHARD (INRIA). I owe much to Najah NAFFAH (BULL) who managed to give me the material support I needed, and offered me the opportunity to pursue this work thru "COCOS". From a technical point of view, my thanks go to Eric GODEFROY (MATRA), Marc SHAPIRO (INRIA), Pierre COINTE (XEROX) for fruitful discussions respectively on type-state checking, iterators and SMALLTALK. Finally, special thanks go to my fiancé Christine and Alain GIBOIN for their constant and irreplaceable support.

REFERENCES

- [1] GOLDBERG, A. and ROBSON, D. *"SMALLTALK-80 : the language and its implementation"* (Addison-Wesley, 1983)
- [2] KRASNER, G. *"SMALLTALK-80 : bits of history, words of advice"* (Addison-Wesley, 1983)
- [3] GOLDBERG, A. *"SMALLTALK-80 : the programming environment"* (Addison-Wesley, 1983)
- [4] LISKOV, B., ATKINSON, R., BLOOM, T., MOSS, E., SCHAFFERT, J.C., SCHEIFLER, R., SNYDER, A. *"CLU"Reference Manual* Lecture Notes in Computer Science #114 (Springer Verlag, 1981)
- [5] LISKOV, B., SNYDER, A., ATKINSON, R., SCHAFFERT, J.C. *"Abstraction Mechanisms in CLU"* Communications of the ACM, Vol 20 #8 (Aug. 1977) pp. 564-576
- [6] LISKOV, B. and SNYDER, A. *"Exception Handling in CLU"* IEEE Trans. on Soft. Eng. (Nov. 1979) pp. 546-558
- [7] STROM, R. E. and YEMINI, S. *"Typestate : A Programming Language Concept for Enhancing Software Reliability"* IEEE Trans. on Soft. Eng., Vol. SE-12, N° 1 (Jan. 86) pp. 157-171
- [8] STROM, R.E. *"Mechanisms for Compile-Time Enforcement of Security"* ACM (July 1983) pp. 276-284
- [9] LIEBERMAN, H. *"A Preview of Act 1"* MIT AI LAB. Memo n° 625 (June 1981)
- [10] THERIAULT, D.G. *"Issues in the Design and Implementation of ACT2"* MIT AI LAB. Technical Report N° 728 (June 1983)
- [11] LIEBERMAN, H. *"Delegation and Inheritance : two mechanisms for sharing knowledge in object-oriented systems"* Journées LOO 85 (Dec. 1985) pp. 79-89
- [12] COINTE, P. *"The ObjVLisp Kernel : a Reflexive Architecture to define a Uniform Object-Oriented System"* LITP (Nov. 1986)
- [13] CANNON, H.I. *"FLAVORS : a non hierarchical approach to object-oriented programming"* MIT AI Memo.
- [14] BOBROW, D. G. and STEFIK, M. *"The LOOPS Manual"* (Xerox - Dec. 1983)
- [15] CURRY, G., BAER, L., LIPKIE, D., LEE, B. *"TRAITS : An approach to Multiple Inheritance Subclassing"* Communications of the ACM (Mar. 1983)
- [16] CURRY, G. A. and AYERS, R. M. *"Experience with TRAITS in the XEROX Star Workstation"* IEEE Transactions on Soft. Eng. - Vol SE.10, N° 5 (Sep. 1984)
- [17] BORRON, H. *"LE_TOOL : a SMALLTALK-like Programming Environment + Reliability and Efficiency"* ESPRIT Project 956 (COCOS) - 200 pages (Nov. 1986)
- [18] BORRON, H. *"LE_TOOL : the typing mechanism"* ESPRIT Project 956 (COCOS) - 60 pages (March 1987)
- [19] SCHAFFERT, C., COOPER, T., BULLIS, B., KILIAN, M., WILPOT, C. *"An Introduction to TREILLIS/OWL"* OOPSLA Conference Proceedings (Sept. 1986)
- [20] Eckart, J.D. *"Iteration and Abstract Data Types"* SIGPLAN Notices V22 #4 (Apr. 1987)
- [21] CHAILLOUX, J., DEVIN, M., DUPONT, F., HULLOT, J.M., SERPETTE, B., VUILLEMIN, J. *"Le_Lisp Version 15.2 - Le Manuel de Référence"* (INRIA - May 1987)

FUNCTIONAL MODELS OF VISUAL PERCEPTION IN OFFICE CONDITIONS

D. Bosman

1. INTRODUCTION

Why do things look as they do?
 Because they are what they are, or
 because we are what we are?
 Harvich and Jameson

In project OS612/1593 the aim is to realise an electronic simulator which provides at the screen of a high resolution CRT monitor look-alike images of real, existing or of not yet existing designs of flat panels of several technologies. The simulator comprises both the hardware and the software. The latter involves

- (i) operating system,
 - (ii) technology and geometric models
 - (iii) experiment methods and
 - (iiii) models for the perception of displayed images.
- This report concerns the place and the task of (iiii).

These models are necessary for three reasons:

- a) the simulator must produce images that "fool the eye" sufficiently to believe that actual flat panel images are seen, notwithstanding the fact that the models are subsets of reality and that the display medium (CRT screen) has properties entirely different from those of the flat panels;
- b) the parameters of conceptual flat panels must be varied to arrive at an optimum solution in design lay-out and cost for a desired quality of the displayed image, as determined by a standard observer. This is a second task of "fooling the eye";
- c) as in practice there are limitations upon the simulator possibilities, these can be extended by adding knowledge of the human vision process to the system. For instance a design optimised for a standard observer; it highly desirable to be tested on a non-standard observer with well defined vision deficiency; it is necessary to be able to change vision parameters at will, one at a time.

Subtask 2.10 of OS1593 is a subset of task (iiii), since the goals defined under a) through c) involves an amount of research and plain work that clearly goes beyond the time and money allotted in OS612/1593.

Nevertheless, subtask 2.10 should provide the experimenter with tools that

are sufficient to relate experimental results to known vision responses and to results of ergonomic experiments involving human subjects (validation of the simulator) and, using the completed and validated simulator, to evaluate the next generation of flat panel displays.

2. THE VISION MODELS

Basis for the partitioning of the visual imaging process is figure 20 [1], reproduced here as figure 1. The visual processing model is mainly based on threshold data, although in the choice of the model supra-threshold conditions can be accommodated once the relevant data are available with sufficient accuracy.

The spatial distance model is thus far split into 4 parts:

- 2.1. Point spreading by a finite aperture ideal lens ($J_o\{\rho\}$ function) and by scatter occurring in the eye ($K_i\{\rho\}$ function), if necessary corrupted by (monochrome) aberrations;
- 2.2. Processing by retinal receptive fields (assumed square instead of circular or elongated, elliptic) depending on average illumination. Field areas are based on integer decimation, although it is possible to add a 2-D interpolation programme allowing some non-integer subsampling;
- 2.3. Transformation of the radiometric data by a logarithmic or power function; at present the log is implemented;
- 2.4. Lateral inhibition simulating retinal response.

An example of this sequence of operations with the exception of 2.2 is given in figure 2 (figure 21,[1]). In this example a perfect radiometric stimulus is provided to the eye (figure 2a). However, in practice this is not the case as shown in figure 3, where the radiometric response of a wrongly driven CRT (not unusual!) is shown for the letters KEES. Obviously such driving format (possible, and also likely to occur, for every conceivable font) deteriorates the image quality. These detrimental summation effects can also happen in perceived images from high resolution displays when eye properties are not properly accounted for.

Especially the kind of effects shown in figure 3 may require an extension of the model steps 2.1 through 2.4 with

- 2.5. A spatial orientation insensitive contour extraction algorithm, the choice of which is based on suggestions in literature by numerous authors that the brain processing entails search and interpretation of contours. Such an algorithm, avoiding many of the artifacts of commonly used contour algorithms, has recently been developed [2] and the software runs on a UNIX V based computer (PCS, Masscomp/I²S). One possible problem present in the models is the coordinates basis which, by nature of the image acquisition and computation constraints, is orthogonal instead of hexagonal. As yet there is no evidence that

the considerable amount of man power and extra computation (execution) time needed to transform to a hexagonal system is warranted. Moreover, the transformation is not completely one-to-one, unless resolution is sacrificed.

2.1. Point spreading in the optical path of the eye.

The response function at the receptor plane of the eye is determined by the lens structure with its finite aperture and aberrations, and by scatter produced by the inhomogenities in the eye body and the mess of blood vessels and nerves located directly in front of the retina. Therefore the proposed model accounts in the first place for diffraction by the pupil (incoherent white or monochromatic light) and for partial scatter by a diffusing plane close to the receptor plane. As yet aberrations have not been included although an aberration algorithm can be cascaded. The simulation result must match the calculated PSF at several pupil sizes as determined from line spread functions [3].

2.1.1 The diffraction effect of incoherent light is modelled by the Fourier transform of a cone (a cone being the result of the convolution of a cylinder produced by the pupil with itself). The area $A(a_2)$ at the retina, the Airy disc [4], is given by (see figure 4).

$$A(a) = 3.67 \frac{r_2^2 \lambda^2}{a} \quad (1a)$$

with r_2 the focal length of the optical system (17 mm), a_2 the pupil area and λ the wavelength of the light. For white light a weighed average is obtained using the luminous efficacy function of the eye $K(\lambda)$.

At a pupil diameter of 2.4 mm, $a_2 = 4.52 \times 10^{-6}$ and $A(a_2) = 70 \cdot 10^{-6} \text{ mm}^2$ for $\lambda = 0.555$ micron; given the receptor density at the centre of the fovea of $2 \cdot 10^5 \text{ mm}^{-2}$ the Airy disc encompasses 14 receptors.

2.1.2 The Airy disc function is rather steep, whereas the PSF calculated from measurements has an exponential skirt. This can be caused by aberrations of the optical system and by scatter in the eye. At small pupil size (supra threshold condition) the latter prevails. The scatter function was developed for a diffusing plane:

$$\text{PSF}(\sigma) = \frac{M \text{ da} \cdot C}{(\ell^2 + x^2)^{1/2}} \quad (1b)$$

where M = exitance of illuminant, da area of illuminant, C a constant, ℓ the distance from the scatter plane to the retina and x the distance from the central position. Preliminary calculations show that ℓ is of the order of 10 micron, see figure 5. With partial scatter, the total PSF experienced at the retina is made up of the weighed sum of the original (Airy disc) PSF and the convolution result of the Airy disc PSF with the scatter PSF.

2.1.3 The total PSF(r), given by weighted combination of diffraction and scatter, calculated for a pupil of 2.4 mm, is analysed below. The diffusing layer, being very close to the retina, is illuminated by a spot with intensity distribution PSF(a₂). The resulting intermediate PSF(i) is given by the convolution of PSF(σ) with PSF(a₂). Consider the diffusion layer as partially transparent, partially diffuse. A fraction a of the illuminating function PSF(a₂) is transmitted undisturbed, a fraction b is scattered, so that

$$\begin{aligned} \text{PSF}(r) &= a.\text{PSF}(a_2) + b.\text{PSF}(i) \\ &= a.\text{PSF}(a_2) + b.\text{PSF}(a_2) * \text{PSF}(\sigma) \end{aligned} \quad (2)$$

where * stands for the convolution operation.

There are only two parameters, a and b, to be varied to match PSF(r) with the PSF(eye) as calculated from the line spread function as determined by Gubisch [3]. Moreover, these two parameters are mutually dependent because of the continuity constraint: without losses due to absorption the total luminous flux impinging on the retina (contained in the volume of PSF(r)) must equal the luminous flux entering the pupil. It follows that

$$\text{Volume PSF}(r) = \text{volume PSF}(a_2) + \text{volume PSF}(i) \quad (3)$$

Expression (3) thus determines both the scaling factors a and b since the form of the contributing PSF's remains constant. The contribution of PSF(a₂) is in figure 6 depicted by the dashed curve, the wider tails thus should be caused by the term b PSF(i). Based on this assumption, the method of calculation of a and b is as follows.

First the volumes of PSF(a₂) and PSF(i) are normalised equal to unity. It follows that the amplitudes of the normalised PSF's are 5.3×10^{-2} units for PSF(a₂) and 2.6×10^{-3} for PSF(i). Invoking (3), with volume ratios of 0.136 and performing the weighted addition yields the cross section of PSF(r) as shown in figure 7a which closely matches that of figure 6. In terms of amplitudes, the scaling factors a and b are 0.12 and 0.88 respectively. The error distribution is given in figure 7b.

2.1.4 The PSF(r) for other pupil diameters.

The Airy disc diameter is inversely proportional to the pupil diameter, the amplitude of PSF(a₂) proportional to pupil diameter squared, so that the volume of PSF(a₂) remains constant. It now remains to determine whether the weight factor for the scatter contribution varies with pupil diameter, i.e. whether the distance ℓ of the hypothetical diffusing plane to the plane of the retina varies with pupil diameter. This is plausible, since the cone solid angle of the focussed light from pupil to retina increases linearly with pupil area so that in volume scattering a larger volume partakes in the scatter process. This effect is offset by the fact that the illumination area of the diffusing layer decreases as d, the diameter of the Airy disc, is inversely proportional to pupil diameter.

The consequence of the change in area of the Airy disc is that the illuminance at the centre of the PSF(a₂) increases linearly with a₂:

the scattered fraction is proportional to the illuminance and to the geometrical properties of the neural tissue which remain constant. If ℓ does not vary (much) then the normalised cross section of the PSF(r) must exhibit a (nearly) constant tail character.

From [3] this seems to be the case for pupil diameters $2 < D < 5$ mm. Therefore, once a's and b's are determined, PSF(r) can be calculated for this range of pupil diameters implying a most simple model.

2.2. Spatial processing in the retina (receptive fields).

At lower retinal illumination the eye counteracts the lack of radiometric intensity, among others, by forming receptive fields where output of neighbouring receptors are combined to yield better discrimination against sources of noise (internal noise, quantum noise). From signal theory [8] it is known that if the pooling follows a proper low-pass filter law (determining weight factors for the contributions of receptors partaking in the pooling) only the eye's resolution is affected and no aliasing occurs. In modelling, two problems arise. Firstly the low pass filter concept involves a convolution process, meaning that the receptive field(s) scan across the image. In actual viewing this may not be the case, since signals from the cognitive process in the brain may selectively control the 2-D linear scanning process. Secondly, no pertinent data are available for the weight factors involved in pooling, and simple uniform averaging does produce unwanted (aliasing!) sidelobes in the 2-D transform to the spatial frequency domain.

There is an elegant way out of this problem, which incurs a particular choice for process 2.4: lateral inhibition, (LI). Using the constant volume summation process as suggested in [5] automatically increases the base area of the LI inversely proportional to retinal illumination. This law may not be completely accurate as already suggested by one of the authors [6], but in a restricted range may be sufficiently correct to be useable in the present stage of the modelling effort.

2.3. Receptor response law.

The receptor response law seems to be a power law [7]

$$f(L) = p(L_1^x - L_0) + q \quad (4)$$

with L_1 foreground luminance, L_0 background luminance, $x \sim 0.3$; p being a sensitivity parameter and q a zero offset determined by internal neural processes. For a long time a logarithmic law was assumed. For macroscopic purposes this still is a sufficient approximation. In the present model of the receptor response the log is implemented:

$$f(L, L_0) = a \log L_1/L_0 + b \quad (5)$$

We realise that this choice implies small errors at low contrast ratios (e.g. for $L_1 \approx L_0$: 30%, $L_1/L_0 \approx 2$: 20%, $L_1/L_0 \approx 3$: 12%, becoming 0% for $L_1/L_0 > 5$), but these small errors affect already small response magnitudes which in itself already are uncertain to the same order of

magnitude of the error. In return the computation effort can be much smaller.

2.4. Lateral inhibition.

As already suggested at the end of 2.2, the constant volume operator or a suitable derivative thereof may be the proper choice for the retinal enhancement algorithm. Its operation is based on a non-linear convolution of the image projected onto the retina with a variable base area PSF with constant volume V , the product of PSF height h and area $A : V = h \cdot A$. As shown in [8], the optical signal to noise ratio (SNR_o) and therefore the local photon catch is constant when

$$C^2 \int h^2(r) dA(r) = E^2(o) \{ \int h(r) dA(r) \}^2 \quad (6)$$

with C a proportionality constant, $E(o)$ local illumination at the centre of the PSF, $h(r)$ and $A(r)$ are the height of the PSF and associated area element at distance r from that centre (PSF circular symmetric).

The consequence of (6) is that for uniform illumination the relation holds

$$\int dA(r) = \frac{C^2}{E(o)^2} \quad (7)$$

The non-linear convolution yields, due to the spreading of the PSF, a response PSF with negative sidelobes even when the original PSF is fully non-negative; therefore may approximate or simulate the lateral inhibition PSF as measured in visuology. This remarkable property also is not very sensitive to the choice of the original PSF.

The response is calculated from $f = E(o) \int h(r) dA(r)$, so that fluctuations propagate according to

$$\left| \frac{\Delta f}{f} \right| = \left| \frac{\Delta E}{E} \right| + \left| \frac{\Delta V}{V} \right| \quad (8)$$

with one contribution due to the illumination fluctuations and one due to volume noise as may happen in computation.

In making the model analogous to the sequence of processes in the eye, one would prefer a spreading function (original PSF) that is most likely in the retina. However, the retinal neuron circuitry is very complicated and every choice will, with the present state of knowledge, be somewhat arbitrary. We implement two functions: one Gaussian (reflecting ignorance) and one (K_o) derived from the potential distribution in a continuous sheet of conducting material where receptor channels are formed by difference amplifiers [9], a natural but still wild assumption. The two spreading functions are shown in figures 8 a and b.

As can be seen, the K_o function is more slender at the centre, but has a wider skirt than the normal distribution; it fits better the measured [3,10] line- and point spread functions. This fit is

important, since the volume of the 2-D PSF under the outskirt is considerable because the radius is large, and therefore its contribution to the total photon catch. Consequently the photon noise becomes less prominent compared to internal noise in the retinal process so that estimation of threshold parameters in the calibration phase may be more accurately reflect internal noise magnitude.

2.5. Contour extraction.

Under the assumption that the symbol identification as it takes place in the brain requires an intermediate step of contour extraction, one needs an algorithm for this purpose with as little artifacts as possible. Many algorithms have been proposed in literature, the better ones still defective at branching nodes and intersections. At UT, in our group, a refinement of the Laplacian based contour extractor has been developed [2] with very little defects, suitable for our purpose.

The problem arises how to assess the derived contours in terms of image quality and/or legibility. The usual way is to scrutinise the contour images for differences with a "perfect" reference and to note obvious defects. It is not known, however, how the brain deals with such defects and, consequently, how to assign confusion probabilities. The project has little room for a learning phase in this respect. It may very well be that the contour interpretation programme in the package ORACLE developed by BAe provides useable answers.

3. FURTHER DEVELOPMENTS

The model described thus far is entirely static, i.e. temporal and spatio-temporal aspects are not included. Moreover, the model should be extended to provide answers on magnitudes of the lower bounds on thresholds (contrast, size) for wanted displayed information and of the upper bounds on thresholds for display artifacts. It is proposed to consider these aspects completely separated, i.e. not to develop a fully integrated model. Further, the spatio-temporal relations are less important in unpaced task performance so that this extremely difficult to simulate aspect could be ignored.

3.1. Within these restrictions, the temporal model would include:

- the Talbot-Plateau law which postulates linearity in perception of intermittent stimuli:

$$\bar{L} = \frac{1}{t} \int_0^t L dt \quad (9)$$

- a modification of the Ferry-Porter law

$$CFF = a \log(1+M)L + b, \quad CFF > 20 \quad (10)$$

where M is the modulation index of the flickering source [11], CFF is critical flicker frequency and a, b are constants;

- the Granit-Harper law [12] which gives a fit for the relation between size of the flickering target and CFF:

$$\text{CFF} = c \log A + d \quad (11)$$

where A is stimulus area, up to 7° diameter; the relation is valid over a range of 7 decades for the fovea. Expressions (3.8) and (3.9) thus can be combined by simple addition of the right hand terms.

In the so obtained model neither the influence of colour contrast nor the effect of surround luminance is taken into account. The latter can be inferred from data [13] in the form of a modifying look-up table (LUT). Under office conditions the relations remain valid for any colour of monochrome stimulation. The effects of colour contrast on flicker perception have not yet been adequately studied within the framework of the project and therefore as yet no model can be proposed. However, available data could be obtained from suitable experts and, again, incorporated in a modifying LUT.

3.2. Contrast and size thresholds must be differentiated to wanted and unwanted phenomena. For studying very local effects, e.g. changes in fonts but also display artifacts such as non-constant sensitivities of display elements, one can use a mathematical model of Blackwell's data which gives a just noticeable contrast difference C_{vth} as function of background luminance L_b and dot size Δx :

$$C_{vth} = \frac{L_f - L_b}{L_b} = 0.18 \{1 + 15L_b^{-0.5}(\Delta x + 0.06)^{-1}\} \{(\Delta x + 0.06)^{-2} + 0.015\} \quad (12)$$

with L_b in cd m^{-2} and Δx the linear angle (in minutes) of the dot. The expression is valid within 10% for office conditions; i.e. $2 < \Delta x < 20$; $3 < L_b < 3000$.

Under office conditions it is highly desirable to operate in supra-threshold conditions for the display elements which constitute the wanted information. In the present state of display technology this requirement can nearly always be met, given the luminance range of electro-optic conversion techniques and the (relatively) low resolution as compared to printing techniques; careful choice of fonts can avoid local (radiometric) peaking in symbols at intersections as depicted in figure 3.

It is known that under supra-threshold conditions the MTF of the eye gradually loses its differentiating character at low spatial frequencies [1, figure 19]. In terms of the point response, it means that the total PSF (including lateral inhibition) no longer has zero total volume as under threshold conditions, but a certain positive value. Simply adding a luminance dependent pedestal is an entirely wrong model; the effect rather suggests the gradual change-over from a constant volume spreading summation technique to a constant volume averaging technique which shows no overshoot like spreading does. Possibly the same result can also be obtained by changing expression (7) to

$$\int dA(r) = \frac{C^2}{E(o)^\ell} \quad (13)$$

with ℓ a luminance dependent parameter, $\ell \leq 2$.

The simulation under supra-threshold conditions still is an object of study.

In the future, when several shades of luminance are possible at higher resolution (del diameter of the order of 1 minute or smaller) threshold conditions may arise for the foreground data, especially at parts of certain artificially generated symbols displayed in complicated (sensitive) fonts. Simulations may show that it then becomes necessary to use part of the shades range to modulate such font refinements in order to exceed very local threshold levels. Thus, modelling of the local size /contrast threshold as function of the local average luminance also is necessary for foreground data.

Unwanted phenomena present in display technologies should preferably remain under the local and global threshold levels. To study their (disturbing, masking) effects a size/contrast threshold model is required. Although it is very desirable to have this in integrated form with the threshold model for foreground data, it may be necessary to divorce them since, for instance, the sensitivity for extended, elongated features like grid lines differs appreciably from that for targets which are small in all directions. The difference would mainly be in the choice of parameters, not in the structure of the model. However, computer experiments still to be performed may show that both situations can be sufficiently modelled by adding a stage where the threshold level depends to a certain degree on (fig. 16 [1]) dip volume instead of dipheight only.

From the above it is clear that our approach to imaging fidelity will predominantly concentrate on processes in the spatial distance domain. We strongly disagree with presently popular concepts based on transformations to the spatial frequency domain (Snyder, Carlson/Cohen) as these are firstly: not valid for non-linear systems like the eye and, secondly: in a linearised regime only give global, statistical measures which are not sensitive enough to model very local phenomena unless they are repetitive over the entire display.

This view puts us in the position where new solutions to the problem must be generated because in this case the literature available cannot be used to advantage.

4. CONCLUSION

In this contribution to the ESPRIT technical week 1987 a few examples of vision models as applicable to office conditions are given; as being planned for use with the display simulator designed and realised in project OS1593. Implemented in an image processor which is also capable to process display design data, post retinal signals of the image to be displayed as defined in

computer display memory are derived. It is expected that the so (off line, hard copy) representation will provide the basis to construct an ordinal rating scale (e.g. bad, legible but not acceptable, acceptable, good) for proposed display designs in several technologies (CRT, LCD, EL, plasma) with project emphasis on LCD and EL.

The models imply that, for the first time, a standard observer is used for display evaluation. Modifying model parameters according to certain vision imperfections and defects would make systematic study of observer variability in the context of display acceptance possible. However, within the constraints of OS1593 the scope of vision modelling is restricted to the standard observer with the goal of obtaining such data as to arrive at the construction of a rating scale.

5. REFERENCES

- [1] Bosman, D., An introduction to functional models of vision. Report 1, subtask 2.10, in ESPRIT project OS1593. April 14, 1987.
- [2] Besuijen, J., Edge detection by combining directional derivatives. In: Proceedings EUSIPCO-86, part 2, pp 887-891, 1986.
- [3] Gubisch, R.W., Optical performance of the human eye, *JOSA*, 57, pp 407-415, 1967.
- [4] Longhurst, R.S., Geometrical and physical optics, Longman Group Ltd, London, 1967.
- [5] Cornsweet, T.N., Yellot, J.I., Intensity - dependent spatial summation, *JOSA-A*, 2, no 10, pp 1769-1786, 1985.
- [6] Yellot, J.I., Photon noise and constant volume operators. Submitted to *JOSA* for publication, March 1987.
- [7] Mackay, D.M., Psychophysics of perceived intensity: a theoretical basis for Fechner's and Steven's law. *Science*, 139, pp 1213-1216, 1963.
- [8] Bosman, D., Bakker, W.H., Conditioning of local image signal to noise ratio. In: Proceedings EUSIPCO-86, part 2, pp 861-865, 1986.
- [9] De Rooij, C., A model of image enhancement based on potential leakage. BSC report 065-2819.
- [10] Westheimer, G., Campbell, F.W. Light distribution in the image formed by the living human eye. *JOSA*, 52, pp 1040-1045, 1962.
- [11] Kelly, D.H., Effects of sharp edges in a flickering field. *JOSA*, 49, pp 730-732, 1959.
- [12] Granit, R., Harper, P., Comparative studies on the peripheral and central retina: II. Synaptic reactions in the eye. *American Journal of Physiology*, 95, pp 211-227, 1930.
- [13] Berger, C., Illumination of surrounding field and flicker fusion frequency with foveal images of different sizes. *Acta Physiol. Scand.*, 28, pp 161-170, 1954.

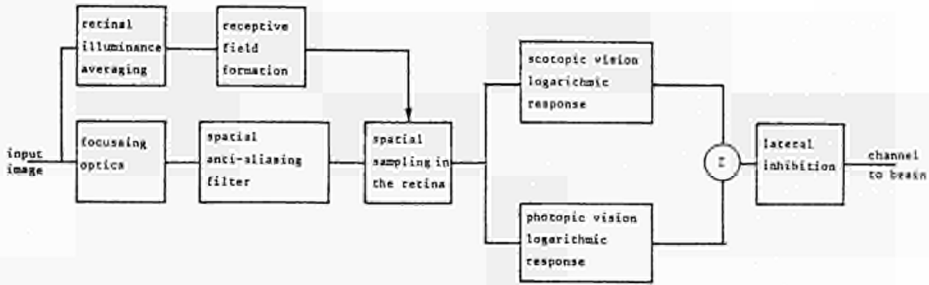


Fig. 1 Blockdiagram of the eye as an image processor.

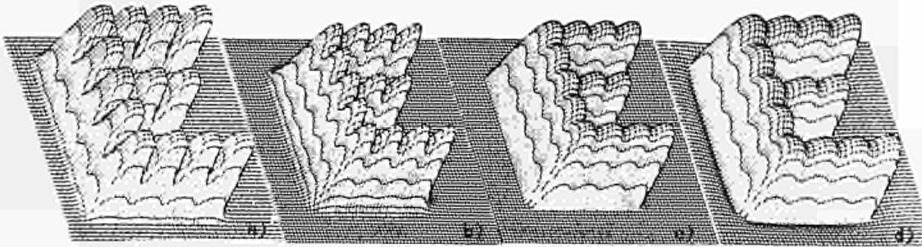


Fig. 2 Sequence of operations in the process of visual acquisition:
 a) letter E as formed on a CRT screen by beam PSF;
 b) the image as projected onto the retina, smeared by eye optics, (geometrical distortion by curved retina omitted)
 c) response of the receptors (log operation);
 d) image after enhancement by lateral inhibition.

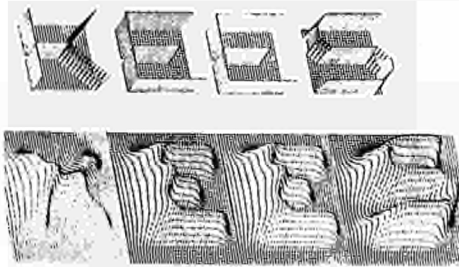


Fig. 3 Radiometric distribution of wrongly addressed fonts, obtained at the screen of a CRT or a high resolution flat panel display.

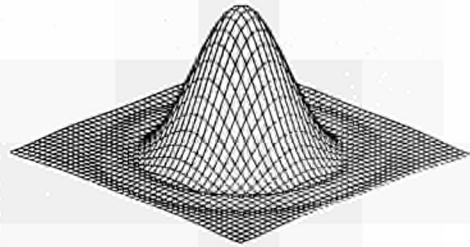


Fig. 4 Bessel function.

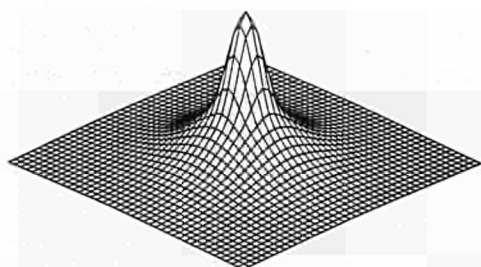


Fig. 5 Scatter function.

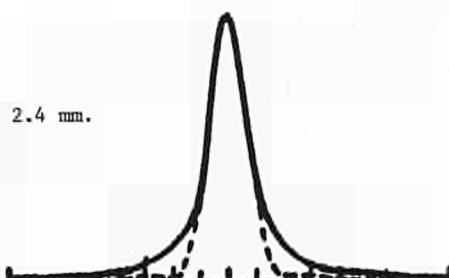


Fig. 6 PSF of eye optics determined by Gabisch [4].

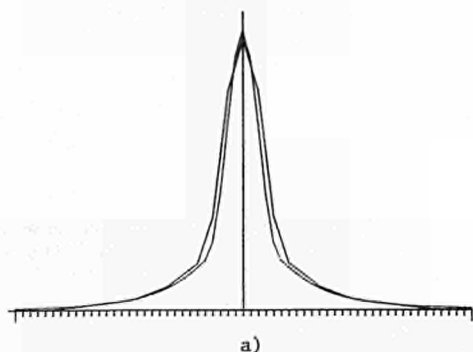


Fig. 7 a Cross sections of measured and simulated PSF for pupil of 2.4 mm.

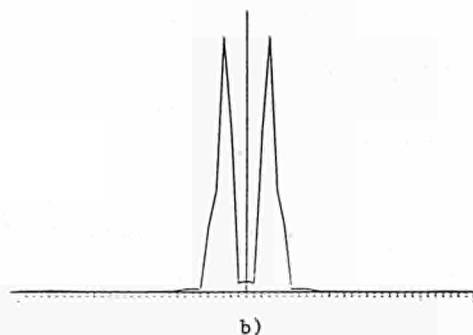


Fig. 7 b Squared error of simulation, magnified 10 times.

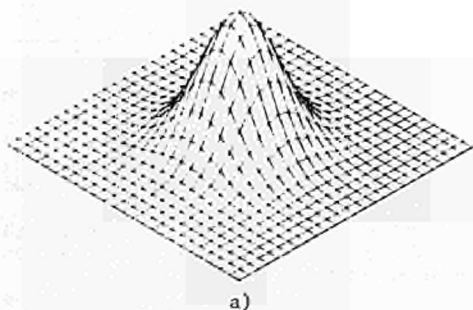
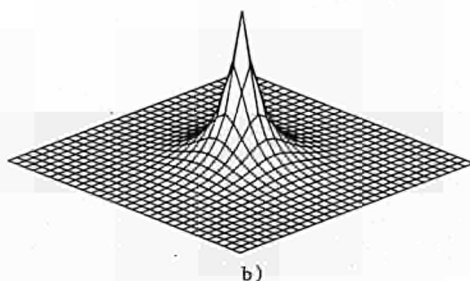


Fig. 8 a Gaussian function.

Fig. 8 b K_0 function.

DISPLAY MODELLING AND ITS SOFTWARE IMPLEMENTATION

Inmaculada Placencia Porrero

Océ-Nederland B.V.
p.o.box 101, 5900 MA Venlo
The Netherlands

This paper concentrates on the modelling of the spatial, temporal and visual characteristics of flat panel display technologies under office working conditions. Display modeling is one of the key parts of the DISSIM machine. The simulation of different technologies, their physical and mathematical models and the software implementation of these models are considered here.

1. INTRODUCTION

1.1. Objectives

The visual characteristics of flat panel display technologies differ considerably from those of the well known cathode ray tube (CRT). The display elements (del) of the CRT have a gaussian distribution of intensity in the emission of light where the electron beam hits the phosphor. Flat panel displays have spatially fixed addressable electrodes overlapping and producing a grid of rectangular dels.

In this project we are aiming to model three different display aspects:

- light distribution of a display element (Point Spread Function)
- switching characteristics of the active layer (rise and decay time)
- colour of the display element.

Models of the different technologies will calculate a description of these three visual aspects for a certain display is described by its engineering parameters. This permits the simulation on a high resolution monitor of the desired display.

1.2. Technologies

The most promising technologies for the development of alphagraphics displays for office use are electroluminescence (EL panels), liquid crystal displays and plasma panels. We are concerned here with displays for computer terminals or word and text processors. They have in common two main requirements, high quality and low cost. Taking this into account and the limited time capacity within this project, EL and LC displays have been chosen for model implementation. Plasma displays could be very easily simulated and included later in DISSIM due to its modular structure.[1]

Ac Thin Film EL displays have been chosen for implementation because of their long life with high luminance (lower in ac powder EL) and matrix addressability (difficult in dc EL). AcTFEL devices are already used in portable computers thanks to the appearance of DMOS high voltage drivers.

The three LCD technologies which have been selected for modelling are the supertwist display[2] the active matrix[3] addressed display and the ferroelectric[4] display. These three LCD types are most likely to be used in office automation applications

1.3. Partners

In this modelling task, there are three partners involved:

- Océ Nederland B.V.
- Cimsa-sintra
- GEC Research

Cimsa-Sintra is manufacturing EL displays while GEC is investigating and producing LCDs. Both provide information about the physical properties, of their respective displays and measurements of the optical and electrical characteristics.

They also supply the project with test devices and the values of their internal parameters. This permits the verification of the performance of the display simulator. They also cooperate with the other partner, Océ, in the development of the models of the different technologies. In addition, Océ is in charge of the software implementation of these models and their integration in the simulator machine

2. SOFTWARE STRUCTURE AND IMPLEMENTATION

The hardware and software of the simulator machine are developed in parallel. Both of them have to be ready for integration one year before the end of the project. The detailed interfaces have been defined during the development of the project. In the mean time another low end hardware system is used to work with the software. These two facts have had a big influence in the conception of the software structure.

The simulation software is design independently of the hardware, defining a Cover Layer around it. This layer is hardware dependant and is built especially to hide it.[5]

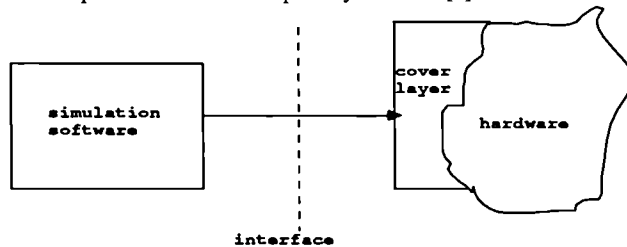


fig.1: software cover layer

The simulation software is then developed following a modular concept. This modularity involves such different aspects as;

- Different technologies. It is possible to work with each technology independently of the others,so that it is possible to implement one without the need for the rest to be developed.

- Detail and complexity of the groups of pixels up to the total screen. This means that first del will be simulated going on with character fonts ending with some alphagraphic facilities.

- Input and output parameters. Considering first basic ones like del shape, rise and decay time and different materials later more complex ones like color and gray levels will be included.

The simulation software is divided in two main parts. The Model Translator and the Simulation Control.The Model Translator has as input the engineering parameters that describe the display device technology

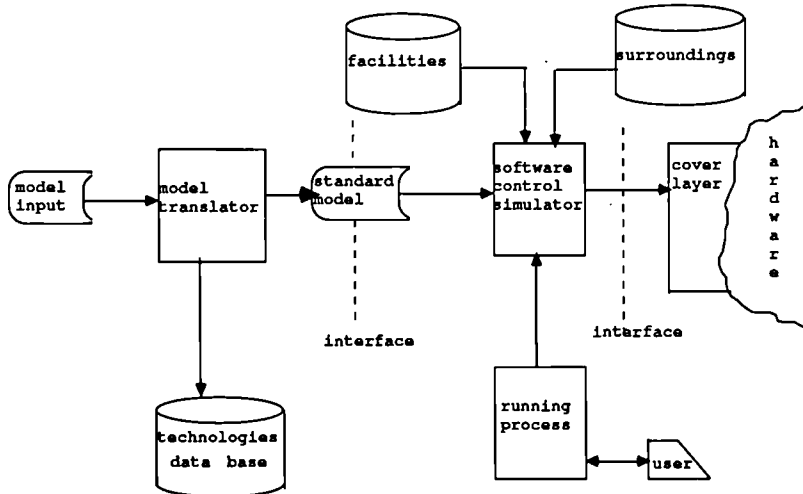


fig.2 software structure

to be simulated. These parameters can be entered interactively or can be stored in the technology database. The Model Translator produces a Standard Model (formatted independently of the technology) defined by a set of standard parameters that are passed to the Simulation Control. The Standard Model contains all the spatio, temporal and visual information of the model. The model obtained is independent of the hardware system. It consists of numerical results, which is very important because it allows us already to obtain information about the model and to test its correctness before we visualize the simulation. The Simulation Control processes this information also considering other characteristics of the simulation session like surrounding reflections, fonts and type of application run by the user. Then it sends these output parameters to the hardware through the Cover Layer. The main function of the Simulation Control is to initialize and control a simulation session.

The Model Translator is then the block of software which concerns the work with models of the display technologies.

The tasks of the Model Translator are:

- Retrieval and storage of new models in the model database.
- Ensuring that the proper input is obtained for each display technology.
- Testing all input values and relations among parameters.
- Calculation of non-input parameters and the Standard Model for each display technology.

Once the model to be entered is determined the Model Translator should take care that the proper parameters are entered and the right functions are called. To accomplish this the Model Translator uses the services of the DISSIM User Interface commanded by the Model Translator manager. The functions to calculate the standard model are also in the Display Technology Modules, grouped for each main technology. [6]

In its basic essence the Model Translator just manages a few tables of parameters, namely the engineering parameters, the intermediate result tables and the Standard Model.

2.1. Model Translator manager

The Model Translator manager allows the user to have a session in which he can perform actions on the models. It is menu driven and makes use of a data base package. The actions are selected via this menu. The following actions are defined:

- select technology
- select model
- select new model
- show model
- modify model
- print model
- store model
- delete model
- translate model
- mock-up model

Within a session the first thing that the user has to choose is the technology, then it is possible to select or enter a model. When a model of another technology has to be used, the user has first to select that technology. At this moment the user is able to enter or load the new model.

The Model Translator manager asks then for a model name (and model id). The unique combination of technology, model name and model id will be checked within the database. This permits it to load an existing model or to enter a new one. If the user wants to enter a model under an existing name, he has the possibility of doing it and deleting the old version.

When a model is entered or selected, all the rest of the actions previously described can be chosen except for the mocking function. This mocking function is provided to permit the visualization of a directly described Standard Model. (That means defining its own Point Spread Functions and Switching characteristics).

Input for the Model Translator manager (and all the other menu input) is performed by a small menu package. The menu package is developed to acquire a uniform way of handling the menus. The layout of the menus is stored in files.

With in a menu distinctions can be made between:

- Text fields: to give information to the user.
- Input fields: for data input.
- Selection fields: to choose an option. Fixed values are returned.

Menus can be made visible to the user by two routines that read a menu file displaying it on the screen by using the curses package and construction table for the selection fields. A selection field can be chosen by positioning the cursor with the help of the arrow keys and pressing the return key. All data entered via the menus and the values of the tables produced as output from the standard models, are stored in the database.

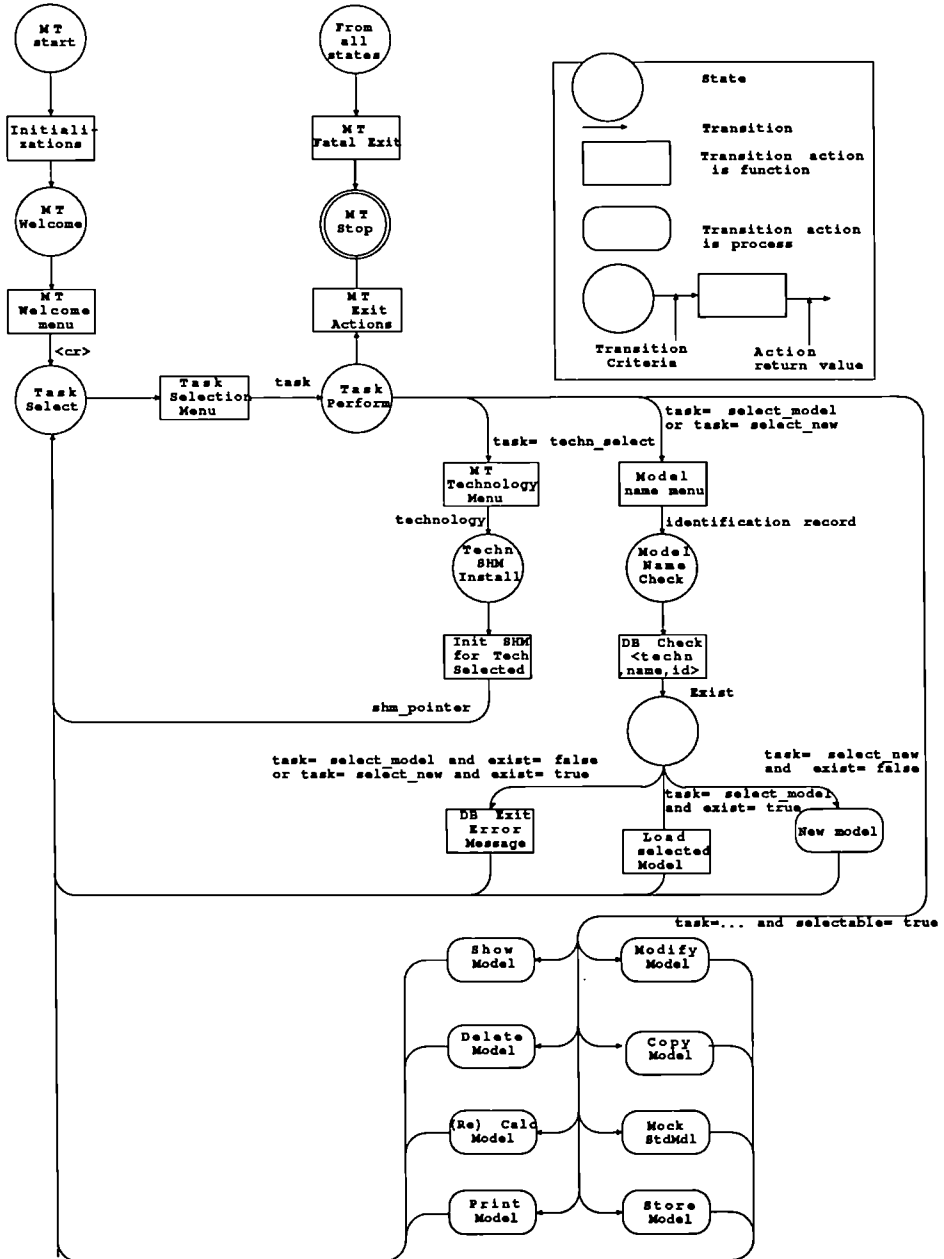


fig.3 model translator manager
(extended state diagram)

2.2. Display Technology Modules

The basic function of this block is to translate a model described with engineer parameters into other parameters that represent the visual performance of the display to simulate. It has certain characteristics that differentiate it from the other software modules: it does not run in real time, it produces numerical results and it is completely hardware independent.

When a model is entered the first action that this module performs is to test the entry checking that the display described is a feasible one, it will reject any incorrect model. If not all the values of the parameters are given it is possible for the model to use default values previously stored.

There are two types of input to the model. One, the user input, is the set of parameters that describes the display technology to be simulated. It is given at the beginning of the translation. The other input concerns all the background information and specifications of the physical and electrical properties of the materials mentioned in the user input. Most of this data is provided when constructing the model but it is also possible for the user to increase the database when entering his data.

The user input is associated in three main groups:

- Device characteristics: contain all the parameters that give the general description of the display panel. First the geometrical description: shape and size. After this a description of the electrical characteristics of the display is specified, parameters like the type of current, voltage range and driving source. Some other general information is also needed like the use of backlight, the operating temperature or the frame rate.

- Display element structure: The geometrical properties of the del are specified here, shape, del and pitch size.

- Panel structure: All the different layers that are forming the basic sandwich structure of the panel are described here. In principle is only necessary to give the number of layers and the type and name of the materials accompanied by their thickness. Immediately there is a search in the database to obtain the optical and electrical characteristics of the layer.

The output is the description of the Standard Model that is used in the software Simulator Control. Together with the rest of the information about surroundings, fonts and application it produces the necessary output to be transmitted to the hardware simulator. All the information about the spatio and temporal characteristics of the simulated display technology are presented in this output. This visual description is hardware independent and uses the same parameters to describe the different technologies.

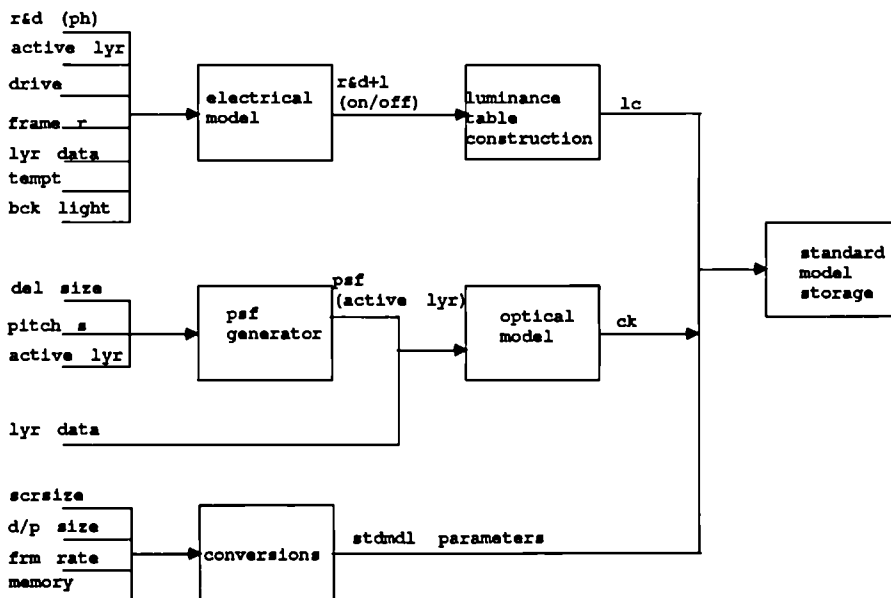


fig.4 display technology modules

The parameters are subdivided in two groups:

-Parameters directly transmitted from the input. Just a small modification in the format is needed to present them to the Simulation Control. No calculations are needed to obtain them (del count, del size, framerate).

-Transformed parameters: Two main data structures are presented here, the first one is the luminance controller table that contains the switching characteristics of the simulated display. The second one is the convolution kernel that contains all the information on the point spread function of the simulated display. Thus the first one takes care of the temporal properties while the second one deals with about the spatial ones.

The display technology modules contain some active blocks that translate the input parameters into the standard output. Inside the blocks the essential physical phenomena occurring in the modelled display are described.

There are three main blocks:

- Calculation of switching characteristics.
- Calculation of point spread function using its model.
- Conversion of input parameters to output parameters.

The switching model has two main parts. The first describes the electrical network and driving sources. This calculates the electrical characteristics of the panel, for instance the influence of the number of dels switched on or off in the light distribution across the panel. The second part is the electrical model of the sandwich which calculates the voltage applied to the active layer. This permits to obtain the contrast ratio of on/off del to be obtained, as well as the delay curves of the system. The next block combines these two results to obtain the luminance controller table.

The point spread function is calculated from the optical behaviour of the active layer and the geometrical description of the dels. Then, following how the light is transformed through the different layers of the display the final light distribution is obtained. This calculation is done for different viewing angles. The output is used to fill the hardware convolution kernel.

The last block is just limited to the transmission of the input parameters to the output adapting their format.

3. ELECTROLUMINESCENCE SIMULATION

3.1. Introduction

Thin film EL devices generally have bright yellow-orange coloured dels on a dark background. This is because the light emitting layer is ZnS:Mn phosphor.

The general structure is a sandwich of two dielectric layers that contain a phosphor layer. Attached to both sides of this multilayer construction there are mutually perpendicular electrodes.

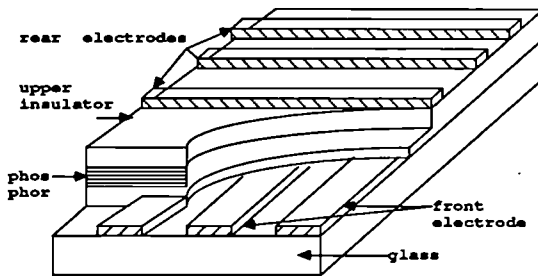


fig.5 el layer structure

Many types of dielectric materials are used but only two types of electrode materials. ITO when transparency is required and otherwise aluminium.

When the proper voltage is applied between the electrodes there is an electrical field created and the phosphor emits light. The basic mechanism of operation of EL devices is poorly understood compared with other display technologies [7].

However it is basically due to hot electron impact ionization of the activator. Chen and Krupka made a first qualitative model of the behaviour of the phosphor but until now there is no quantitative model that predicts or allows the calculation of the phosphor efficiency.

3.2. Point Spread Function

Phosphors in acTFEL devices emit light in a uniform way (there are no optically resolvable sources of light) only from the area where the two electrodes overlap. That light is travelling through the successive thin layers before finally emerging from the glass into air. Light is then refracted and different optical phenomena have to be considered to be able to explain the light distribution. This light distribution (PSF) has been measured at different angles. A del is quite a perfect block when looking perpendicular at the screen. On increasing the viewing angle the block bends losing its symmetry mainly due to internal reflections.

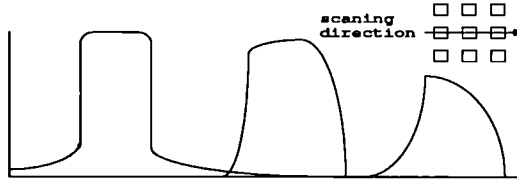


fig.6 EL PSF at 0° 30° 50°

The back electrode (AL) is acting as a perfect mirror duplicating the light output per del. It is observable that light emitted by one del reaches the neighbouring ones. PSF are overlapping each other in the space. This is taken into account in the DISSIM machine, a convolver is built to reproduce these effects.

3.3. Content of the model

The general aim of this model is to obtain the representation of the point spread function for thin film electroluminescence technology. It contains two parts:

- Brightness versus voltage characteristic. This calculates of the light emitted in the phosphor layer when a certain voltage is applied.
- PSF. Transformates that light taking into account the optical behaviour of the layers.

The temporal characteristics will not be included because the rise and decay time of the phosphor is limited by that of the phosphor of the monitor, usually these times are of the same order of magnitude and so it makes no sense to consider them.

3.3.1. B/V characteristic

To obtain the PSF of the technology a first block is developed in which the voltage wave provided to the sandwich layers containing the phosphor is converted into luminance. This is done following the model of P.M. Alt [8]. This model considers the insulator layers of the device as perfect capacitors, the phosphor layer also behaves as a capacitor below a threshold voltage, above threshold it behaves as a resistor with a power consumption proportional to the light emitted in the device.

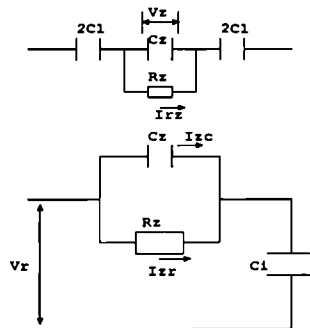


fig.7 EL model

The electrical parameters of the layers; dielectric constant capacitance, breakdown voltage, efficiency of the phosphor and the physical parameters, size, thickness and some general parameters of the entire device, size, resolution, frequency of excitation are requested. From them the following steps are followed to obtain to the B/V curve.

1 - Calculate the threshold voltage of the device.

From the picture it is seen that

$$V_a = V_i + V_z$$

And the voltage is capacitively divided

$$V_{zi} = \frac{C_1}{C_1 + C_z} V_a$$

$$V_{ii} = \frac{C_z}{C_1 + C_z} V_a$$

where the capacitances are calculated with the electrical data and geometry of the materials.

Knowing that $V_{ap} = E_i d_i + E_z d_z$ and that the boundary conditions of Maxwell's equations are imposed on the interface between the layers $E_i E_i = E_z E_z$. And having the threshold voltage of the phosphor V_{thp} it is then possible to obtain the threshold voltage of the device V_{thap} .

$$V_{thap} = V_{thp} \frac{E_i d_i + E_z d_z}{E_z d_z}$$

2 - Calculate the peak voltage of the device.

Considering that the electric displacement is continuous just below threshold it is then clear that Q_z (maximum phosphor displacement) is its value for the insulator at threshold. Then all the voltage above threshold is transferred to the insulator adding charge up the maximum Q_z . So that the maximum voltage excursion above threshold is:

$$V_a - V_{ath} = \frac{(Q_i - Q_z)}{C_i}$$

From what V_a is easily calculated.

3 - Calculate the luminance for the voltages between those two values.

The two former values of the voltage are the ones which correspond to the initial light emission and the breakdown of the sandwich. To calculate the luminance corresponding to these values is necessary to know that this is proportional to the real charge transferred to the resistive branch of the phosphor layer.

$$L = 2nf Q_{ph} V_{thp}$$

n = internal experimental efficiency of the phosphor

f = driving frequency

V_{thp} = threshold voltage of the phosphor.

To calculate the value of Q_{ph} we know that.

$$p_{hr} = i_i - i_{phc}$$

and

$$V_{ph} = - \Delta V_i$$

$$Q_i = C(\Delta V_i) = - C \Delta V_2$$

$$Q_{phc} = C_{ph} (\Delta V_{ph})$$

$$Q_{phr} = Q_I - Q_{phc} = -(C_I + C_{ph}) \Delta V_{ph}$$

and knowing that the phosphor voltage increment is capacitively related to the applied voltage.

$$V_{ph} = \frac{-C_I}{C_I + C_{ph}} (V_a - V_{ath})$$

which leads to the expression

$$Q_{phr} = C_I (V_a + V_{ath})$$

and going back to the luminance

$$L = 4nf C_I (V_a - V_{ath}) V_{phth}$$

4 - Calculate the luminance per display element.

To obtain it, the luminance is simply divided by the area of one del. The luminance is a function of the frequency of excitation, efficiency of the phosphor, thickness of the materials, their capacitance and threshold voltage of the phosphor and of the device.

3.3.2. Optical behaviour of the layers

To determine the PSF on the surface of the display it is necessary to describe the transformations of the volume of light as it goes through the layers. Two characteristics of this light that help to determine the PSF. Firstly there are no optically resolvable sources of light, they are a uniform haze of evenly distributed unresolvable light sources, secondly the fringe field or light-emission fringe is insignificant. Light is emitted only from the area where the two electrodes overlap.

The problem can be abstracted considering a surface that is emitting light and it is double due to the back aluminium which behaves as a perfect reflector. It is necessary to calculate the Brewster angle for the different layers obtaining a cone of light of about 20°. To obtain this value, the interfaces where the index of refraction of the incident medium is larger than the one of the transmitted medium are taken and the angle for which $\sin \theta_i = n_{12}$ is calculated for all these layers. Then it is calculated back to the first layer.

For every punctual source of light then the reflectivity and transmittivity are calculated, considering that for the values of angle (20° or smaller) the intermediate layers are practically transparent, thus contributing mostly to the refraction of the ray than to the amount of light transmitted. Also internal multiple reflections are calculated to verify their influence in the shape of the PSF.[9]

The PSF are then represented as a function of the transmittivity depending on the absolute position within the display element. The results are different for the different scanning angles.

4. LIQUID CRYSTAL SIMULATION

4.1 Introduction

In the area of office automation LCDs offer several advantages over CRTs, in particular flatness, low power consumption and excellent legibility in high ambient lighting conditions. All LCDs consist of similar components [10]. The liquid crystal layer itself is sandwiched between glass cell walls which are coated on their inner sides with indium tin oxide (ITO) electrodes and polymer alignment layers. A polarising film is laminated onto the outer side of each glass wall. A reflective or transreflective (partially transmissive) layer can be laminated onto one side of the display. The differences between the various forms LCD are mainly in the type of liquid crystal used, the configuration of the liquid crystal layers, or the method of applying voltages to this liquid crystal layer.

4.2 LCD Modelling

The procedure for modelling an LCD falls into three main sections. Firstly the liquid crystal layer of the display is modelled by calculating the static configuration of the liquid crystal director (the mean orientation of the rod shaped modules). Then the change in this director configuration with time is modelled. Finally the optical properties of the display are modelled and the transmission calculated for each director configuration. The output from one section of the modelling forms part of the input for the next section.

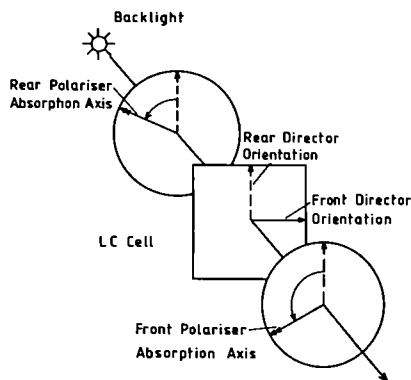


fig. 8 the definition of the polariser orientations

4.2.1 Static Director Modelling

In order to calculate the liquid crystal director configuration for a display at constant voltage the Helmholtz free energy is minimised. The Helmholtz free energy is the sum of electrostatic and strain free energies of the liquid crystal and so is dependent upon the two dielectric and three elastic constants of the material. In addition to these material constants the modelling procedure requires the values of the twist and tilt angles imposed by the surface alignment layers, the natural pitch of the helical liquid crystal and the voltage applied to the display.

Two approaches have been used to compute the minimum energy director configuration. In the first approach, developed by Berreman [11], the conditions for minimum Helmholtz free energy are expressed as pair of differential equations. Repeated integration of these equations for a display iterates towards the required director configuration. More recently the DIMOS [12] LCD modelling package has been evaluated. This package models the liquid crystal layer as a stack of uniform sublayers. The Helmholtz energy of the stack is then minimised by using a Ritz method. Both approaches produce an output which consists of the liquid crystal director orientation as a function of its position within the liquid crystal layer.

4.2.2 Dynamic Director Modelling

Using the director configuration at constant voltage as a starting point, the dynamic change in this configuration can be calculated for any change in applied voltage. In addition to the input parameters required for modelling the static director configuration the five viscosity coefficients of the liquid crystal and the change in applied voltage are needed for the dynamic modelling. In the Berreman approach to liquid crystal dynamics [13] the Erickson [14] and Leslie [15] hydrodynamic equations are solved numerically giving the director orientations at any required set of time intervals. The DIMOS version of the program has not yet been evaluated.

4.2.3 Modelling of Display Optics

The calculation of the propagation of light through the liquid crystal layer forms the basis of the optical modelling of the display. This requires the director configurations obtained in the earlier modelling and the wavelength dependent refractive indices of the liquid crystal. In addition the refractive indices and thicknesses of the polyimide, indium tin oxide, glass and polarisers used in the display are needed.

In the Berreman approach the liquid crystal layer is modelled as 240 uniform sublayers. The transmission of the light through this stack of sublayers is calculated using a four by four matrix formulation [16]. The DIMOS package also use the four by four matrix approach. It models the liquid crystal layer as 80 sublayers but achieves improved accuracy in its numerical calculations. Both packages calculate the transmission through the other layers of the display in similar manner to the liquid crystal layer giving the total transmission of the display for a particular wavelength of light. Colourimetric data is obtained by repeating the transmission calculation over a range of wavelengths spanning the visible spectrum and combining this with the spectral distribution of the ambient illumination.

4.3. Initial Results

A development - version of the suite of LCD modelling programs has been tested against an experimentally measured supertwist display. In order to simplify the initial modelling reflection effects within the display have been ignored. Therefore the modelled value for display transmission is likely to be higher than the experimentally measured value.

For an LCD the contrast ratio of the display, the ratio of the brightness (tristimulus Y value [17]) of 'off and 'on' pixels, is dependent upon the orientation of the polarising layers with respect to the liquid crystal layer. The contrast ratio of a single pixel has been modelled and measured as a function of polariser orientation for a supertwist display. The display was configured for transmissive illumination using a standard, D65 illuminant as the backlight. The front polariser was the layer closest to an observer. The orientations of the polarisers were defined by the angles between the absorption axes of the polariser layers and the director at the rear of the liquid crystal layer (figure 8.) Figures 9 and 10 show the modelled and measured contrast contours, respectively.

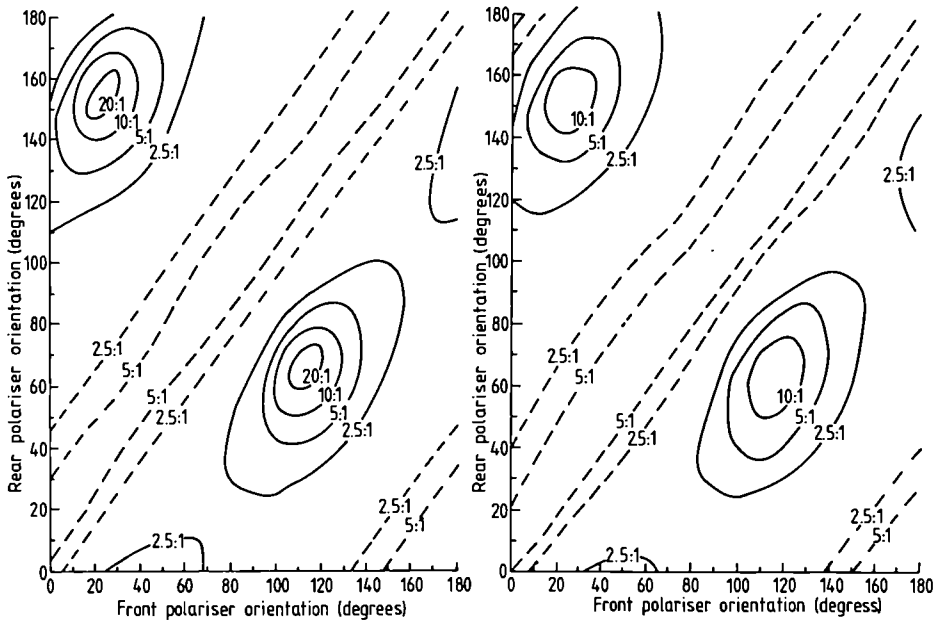


fig.9 Modelled single pixel contrast contours. fig.10 Measured single pixel contrast contours.

Solid contours indicate that the 'off' state is bright. Dashed contours indicate that the 'on' state is bright.

The results demonstrate that there is good agreement between modelling and experiment with respect to the shape of the contrast contours and the positions of the maxima. The absolute values of the modelled contrast ratios tend to be higher than the experimental values. However this general discrepancy is now thought to be due to the approximations used in the development version of the model.

The LCD modelling suite has also been used to explain an experimentally observed effect in supertwist displays. It was found that when the illumination of a supertwist display was changed from ambient daylight to a green electro luminescent (EL) backlight, a pronounced shift occurred in the display colour. The size of this shift was dependent upon the thickness of the liquid crystal layer. When the displays were modelled this effect was found to result from the difference in spectral distribution of the D65 'daylight' illumination and the EL backlight. Figure 4 shows how the modelling programs were then used to find the product of liquid crystal layer spacing and birefringence (Δnd) which minimised this colour shift.

The results shown in figure 11 indicate that for the present display parameters a cell spacing birefringence product of 5.5 μm would only produce a change in colour saturation when the illumination was changed from D65 to a green EL panel.

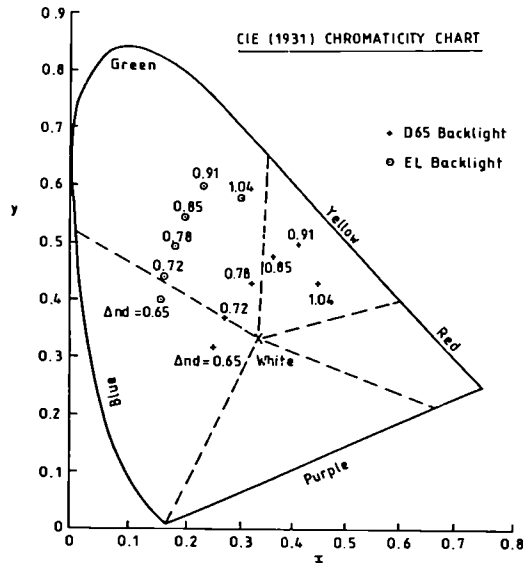


fig. 11 The shift in chromaticity coordinates resulting from a change between D65 and green EL illumination. Cell spacing-birefringence product (Δnd) varying.

5.CONCLUSIONS

The models here presented are implemented to be visualised in the simulator machine. Although the description of the models is independent of the hardware, the accuracy of the images is limited by certain of its characteristics: resolution, speed, colour. The software model can consider more accurate values and they are calculated when some meaningful numerical results are expected to be obtained. The models contain the basic phenomena of the technology and they can be easily modular extended to include new details.

The programs required for the numerical modelling of LCDs are well developed. The next stage of the LCD modelling task will be to implement these programs in the simulator machine. An initial comparison between the output of the model and experimental measurements has demonstrated the encouraging accuracy of the programs. Already the LCD modelling work has produced useful results for LCD manufactures.

ACKNOWLEDGEMENTS

The author wishes to thank Mr P. A. Gass and Dr. M. C. K. Wiltshire of GEC Research for providing results from the modelling and measurement of supertwist LCDs. Also to Mr.S.G.H. van OS for his help with the computing work.

REFERENCES

- [1] Tannas, L.E. Flat-panel displays and CRTs. Van Nostrand Reinhold Company.
- [2] Raynes, E.P. and Waters, C.M., Displays 8 (1987) 59
- [3] Kaneko, E., Liquid Crystal TV Displays (D.Reidal, Dordrecht, 1987)
- [4] Escher, C., Kontakte 2 (1986) 3
- [5] Van Huijstee, H., Os, S. van, Placencia Porrero, I. Listings of programs and other internal reports. ref. OS612/1593. Océ Nederland B.V.
- [6] Parnas, D.L. On the criteria to be used in decomposing systems into modules, Comm. ACM, vol.15 n° 12, dec.72.
- [7] C.N. King Thin film electroluminescent displays. SID seminar 4.1, 1982.

- [8] P.M. Alt Thin film electroluminescent displays: device characteristics and performance. Proceedings of the SID, Vol. 25/2, 1984.
- [9] N.A. Vlasenko Investigation of interference effects in TFEL zNs. Mn. Optics and spectroscopy, Vol. 28, 1978.
- [10] Bahadur B., Mol. Cryst. Liq. Cryst. 109 (1984) 3.
- [11] Berreman, D.W., Phil. Trans. R. Soc. Lond. A 309 (1983) 203.
- [12] Becker, M.E., Japan Display 86 (1986) 300.
- [13] Berreman., D.W., J. Appl. Phys 46 (1975) 3746.
- [14] Ericksen, J.L., Trans. Soc. Rheol. 5 (1961) 23
- [15] Leslie, F.M., Arch Ration. Mech. Anal. 28 (1968) 265.
- [16] Berreman, D.W., J. Opt. Soc. Am. 62 (1972) 502.
- [17] Chamberlin, G.J. and Chamberlin D.G., Colour (Heyden, London, 1980)

Project No. 612

ARCHITECTURE REQUIREMENTS AND SPECIFICATIONS DISPLAY SIMULATOR

Andries E. van der Meulen

Océ-Nederland B.V.
p.o.box 101, 5900 MA Venlo
The Netherlands

The display simulator project is concerned with the development of a flat panel display simulator facility. It comprises dedicated high speed hardware, a modified state of the art graphic processing system, a digitally controlled high resolution monitor, models of flat panel display technologies, as well as models of visual perception.

This paper will mainly concentrate on the ultimate display simulator machine.

1. INTRODUCTION

The aim of the DISSIM project is to identify user requirements for flat panel display developments. Those requirements will be identified from ergonomic experiments, running on a real time interactive flat panel display simulator. The experiments will be designed in the context of office work conditions. The different subareas of the project are clearly expressed by the project objectives:

- To design and build a real time flat panel display simulator. This will be a tool for the flat panel display designer in which design parameters can be optimised in an interactive way by experimenting.
 - To identify, from "ergonomic" experiments, designed to simulate prolonged office work conditions, user requirements for flat panel display developments.
 - To give a lead to European display manufacturers in designing user acceptable flatpanel displays, both by using the simulator tool and by increasing the cooperation between manufacturers, ergonomic experts and office system experts.
 - To describe an engineering model of the Visual System Technology Interface (VSTI) wherein the visual interactions between the spatial and temporal properties of stimuli are taken into account.[1]
- The current DISSIM project is basically a modification and extension of the former OS612 project which started in october 1984, after an initial troublesome start the projectscope was broadened and new partners joined the initial consortium in october 1986. The main changes and additions of the current OS 612/ OS 1593 project with regard to the initial OS 612 project are:

- Ergonomic experiments, which were decreased in resources during the 1984 negotiationphase, are extended, as there was a consensus between partners, review experts and the Commission that this area of investigations is one the focus points in the project.
- The development of dedicated high speed hardware, with computation speeds up to 20 Giga operations /second, requested considerably more effort than the original project plan foresaw, substantial changes had to be made in this part.
- The projectmanagement changed from an University to an industrial party
- To broaden the projectscope and to ensure more reliable data to develop and validate the display models an involvement of display manufacturers appeared to be a necessity. the current cooperation between display manufacturers, ergonomic experts and office system manufacturers forms a sound base for the extended DISSIM project.[2]

Today the DISSIM consortium consists of the following partners and main workpackages:

- Océ-Nederland B.V. (NL)
 - Displaymodel development and implementation
 - Projectmanagement
- Twente University (NL)
 - Ergonomic experiments, vision modelling and displaycover modelling
 - Simulator control software

- Cimsa-Sintra S.A. (F)
Hostcomputer and image processor
Participation in EL modelling
- Myfra S.A (F)
Dedicated high speed processing hardware:
convolutions, switching characteristic processing up to 20 Giga ops/sec
- Barco Ind (B)
High performance digital controlled monitor
- GEC Research (UK)
Participation in LCD modelling

This paper will mainly concentrate on the display simulator machine, another DISSIM presentation during this conference will concentrate on display modelling and its software implementation.

1.1 System overview

An overview of the total system is given in figure 1. The various blocks of this figure will be commented in the paragraphs below.

1.2 Dissim manager

This system part will control the overall simulation, basically this means that over here we find:

- simulation control
- database
with models of display technologies, parameters of materials, models of displaycovers, models of visual perception, etc.
- application control
programs to run various experiments using office applications as for instance wordprocessing.

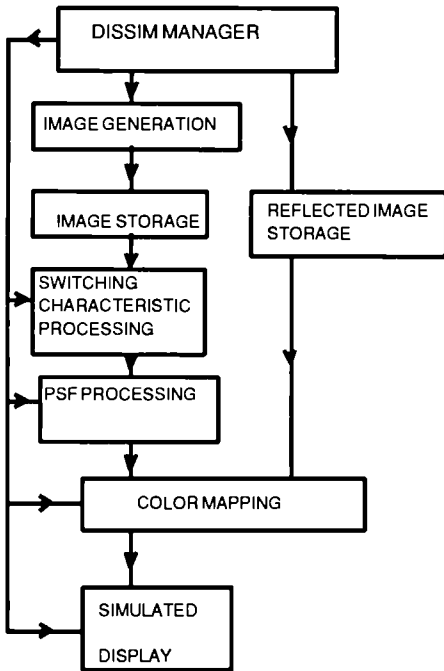


Fig. 1 Systemoverview

1.3 Image generation

Using a set of graphics processors, running a certain simulation, application programs have to translate their output to the display device. In this part, making use of the knowledge of how many pixels on the screen will be used for one simulated del (display element) bitmaps for fonts, graphic primitives have to be generated.

1.4 Image storage

A generated image will be stored in bitmapped memory, acting as a kind of buffer between the image generation part of the machine and the dedicated high speed hardware.

1.5 Reflected image storage

Reflected images, as they might appear on simulated displays, are prerecorded for different technologies, display covers and simulations. In advance of simulation they will be loaded in bitmapped memory.

1.6 Switching characteristic processing

The switching behaviour of different display technologies is one of the most important visual characteristics. For instance working in an interactive way with multiplexed twisted nematic Liquid Crystal Displays, mouse and cursor manipulation, is quite impossible today. Caused by the slow response times of those devices.

This part of the system will calculate in real time the switching behaviour of those devices in order to simulate in real time .

1.7 PSF processing

Point Spread Functions (PSF) of displays are another important visual characteristic. Different technologies show different point spread functions. For instance in the case of a cathode ray tube the PSF is approximately gaussian. Today's LCD and EL devices prove to have approximately square like PSF's. In the case of plasma displays we deal with a crater like PSF. Apart from the technology, the PSF is dependent on the viewing angle.

In the simulations this will be visualised by building up a simulated Display Element (DEL) from a matrix of pixels on the screen. The PSF of a particular device will be build up from a matrix of gaussian (CRT) PSF's. In this part of the system calculation speeds in the order of 20 Giga operations/second are required.

1.8 Color mapping

This part of the system will translate the limited number of possible colors from both, the generated and reflected images, in colors stemming from an extended color map. Apart from that the reflected and generated images will be combined in this system part.

1.9 Simulated display

At the end of the system chain, the simulated display will be visualised at a high performance digital controlled monitor. This monitor system, based on CRT technology, will have the possibility to interchange a full color monitor for a monochrome version in simulations where resolution is more important.

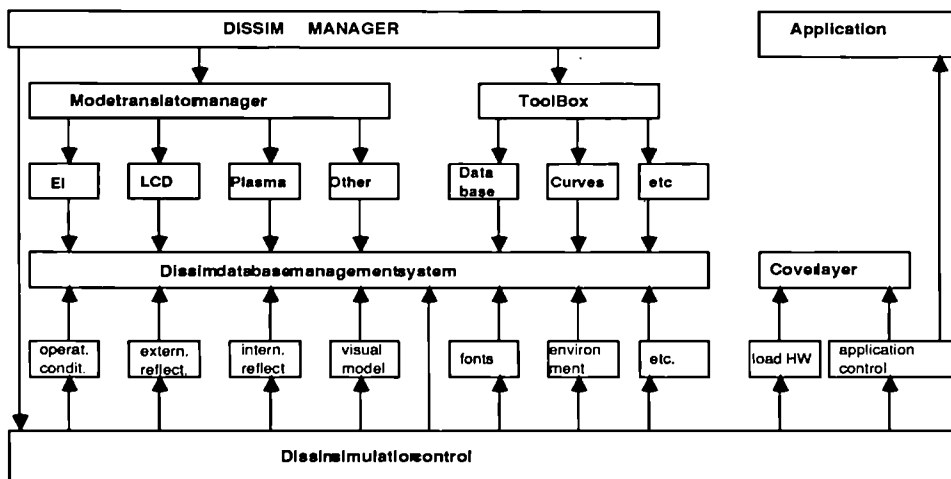


Fig. 1A Dissim Module dependency chart

2.0 FUNCTIONAL ARCHITECTURE

Figure 1A shows the dependency between the different software modules of the simulator machine. In figure 2 the architecture of the more or less generic part of the simulator machine is drawn. It consists of a VME/UNIX 5.2/68010 host computer, interfaced with a slightly modified Triade 60 graphic processor, both provided by Cimsa-Sintra. The modules that will be described in the paragraph 2.1 are an abstract of report OS 612 ref. 8622.080. Here their functional description will be specified, at this moment some are only partially implemented (simulation control and DISSIM manager) and others are not even started yet (data base, application control).[3]

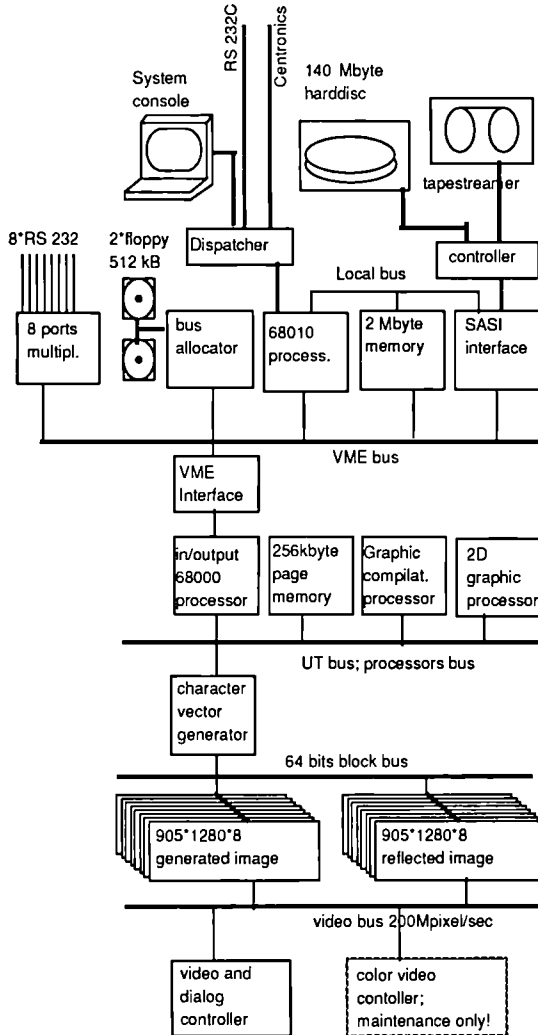


Fig.2DLX/TRIADE60configuration

Fig. 2 DLX/TRIADE 60 configuration

2.1 Dissim manager

The Dissim manager will switch between the services that the manager provides to the user. It will give the simulator the appearance of a turnkey system, providing the user with a menu driven JCL, using the services provided by the Dissim user interface manager (DUIM). The services that are provided are at least; model translation, simulation control, and several tools (to fill the data base, to enter images, etc).[4]

2.1.1 Model translator

Is the package that basically provides the services to enter a display model by its engineer parameters description, load models from data base and calculate its spatio-temporal characteristics into a standard format.

2.1.1.a Database

As a general database package the Empress relational data base system will be used. It will provide services to store, delete, maintain, and retrieve display models, data, images and fonts, as well as all the rest of the information not considered here and that has to be stored in the data base.

2.1.2 Simulation control

The Simulation Control Manager initializes and controls a simulation session. The initialization results in tables that can be loaded in HW for static or dynamic simulation by using the services provided by the cover layer.

During static simulation the user is able to change one or more of the parameters tables, by loading them into hardware. For dynamic simulation it is only possible to switch among the preloaded tables. The latter simulation is based on restricted properties and requires the use of the application environment.

2.1.2.a Application control

The application controller routines provide the service of converting the application program data into the data that is needed by the cover layer to be transmitted to the hardware. The application controller creates an application environment which permits the use of the Barco monitor as if it was a normal display terminal connected to the DLX, visualizing the output of the application program running.

2.2 Image generation

Running simulations "under office workconditions" implies from the application point of view things as: document generation, different fonts and business graphics. It is obvious that in those kind of applications handling speed (interactivity), plays an important rôle.

2.2.1 Character generation

Speedperformance is the most important item here to discuss. The Triade 60 standard character generator is capable to generate characters of 8*8 pixels or multiples of this, up to a maximum of 32*32. Basically a memory accesscycle always address blocks of 8*8 pixels in the bitmapped image storage, either in read, write or read-modify-write mode. In principle the character generator can calculate and transfer to image storage memory a 8*8 pixels cell in 320 ns. . The regeneration of an image (905*1280 pixels) using the charactergenerator would cost approximately 6.5 ms, well below the refreshrate of the systems monitor, so far no problem. The problems are introduced by the fact that DELS are built up from PELS; a typical 4*4 DEL will require only 1 PEL switched on in the bitmap of the TRIADE. Not counting for the needed compilation time in the TRIADE, see chapter 2.2.2

2.2.2 Character compilation

As the dels are built up from pixels it is quite obvious that there will be simulations in which even a 32*32 pixels matrix is to small, however, realise that in those kind of simulations we are dealing with less than 32*40 characters on a simulated screen!

Not using the standard charactergenerator means using the graphic compilations processor. As characters have to be compiled and transferred to image storage memory this process takes more time.

We have to make distinction between two possibilities. In the first one we are dealing with bilevel characters in the second case individual DELS in the characters can have different gray/color levels, i.e. gray level or soft fonts.

Bilevel characters

The fastest process that Cimsa-Sintra provides today, the command CPNT, takes about 11µs per on del . For a simulation in which 4*4 DELS are used and the fontmatrix is 7*9 DELS, the typical number of "on-switched" DELS in a character will be 20. So, for an average screen $35*36*20*11\mu\text{sec}=0.27$ sec will be needed. Together with the scrolling/rolling mode as outlined in chapter 2.3 this implies that the scrolling performance is satisfying.

Soft fonts

In this case the required time will be approximately a factor 3 over the bilevel time. This will be further specified.

2.3 Generated image storage

Basically the configuration will be equipped with bitmapped memory of 1024*2048*8. Due to the used scanformat this will be configured as 905*1280*8. Depending on the particular simulation several possibilities for the use of the bitmap have to be considered:

-switching characteristic simulation;
 in this case static delts are bilevel, so no greylevel or color capabilities are possible. The simulator will only use one plane out of the eight available planes in bitmapped memory. As an additional feature in this mode 128 tables with switching characteristic tables are resident in the machine which give the possibility to run head movement experiments in which the tables have to be switched in the frame retrace time.

-color/grey level simulation;
 in this case no switching characteristic simulation is possible, so no rise and decay times, but 256 greylevels or colors are available for every (static) del.

-combination mode
 in this mode all possible combinations between the first two simulations are possible. In practice this means that in total 8 bits are available for every pixel, they have to be divided between color/gray levels and the number of available switching characteristic table, see also paragraph 2.5

Scrolling and rolling will be performed by changing the readout start address of the generated image storage, this is possible in increments of 8 pixels. Using an eight pixels increment every frame retrace means a scrolling speed of about 1.5 second per screen.

2.4 Reflected image handling

For the reflected image the same amount of memory is available as it is for the generated image; 905*1280*8. Prerecorded reflected images are stored on a tape streamer and/or hard disk. In most simulations reflected images can be downloaded before the actual simulation run starts, although in the case of headmovement simulations, (using different viewing angles), it might be necessary to download images during simulation runs.

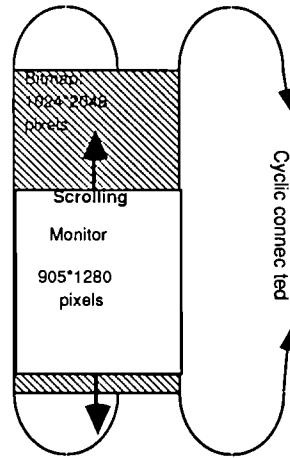


Fig. 3 Scrolling in portrait mode through the bitmap

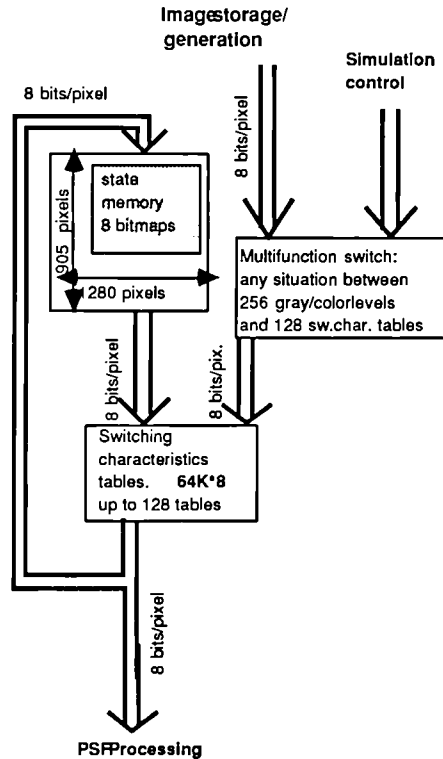


Fig. 4 Switching characteristic processing

The speedperformance of reflected image transfer from hard disk to reflected image memory for a 905*1280*8 image will take 4.5 sec. as a maximum.

2.5 Switching characteristic processing

Figure 4 explains the functional architecture of this part of the simulator machine. Basically it is a matrix of 905*1280 state machines, one for every pixel. The switching characteristics, rise and decay times, are loaded in a kind of look up table in advance of simulation runs. This table is called switching characteristic table. In fact it is a configurable table, downloaded in advance of a simulation, and selectable and configurable during a simulation in the frame retrace time. The basic idea behind a configurable table is that dependent on the simulation mode, see also paragraph 2.3, a choice has to be made how many gray/color levels will be used:

In the case of a bilevel simulation, only one bit/pixel is needed, in this case the table size is 512 byte and due to the amount of available memory (64k *8) it is possible to have 128 tables available in this part of the machine, which is useful in the case of headmovement experiments.

In the case of gray/color level simulations, 8 bits/pixels are used. The table size in this case will increase to 64kByte. This implies that only one table can be resident in the machine.

Between those two extremes any possibility can be configured.[5]

no of color/gray levels	no of poss. view. angles	size of table
2	128	512 Byte
4	64	1 kByte
8	32	2 kByte
16	16	4 kByte
32	8	8 kByte
64	4	16 kByte
128	2	32 kByte
256	1	64 kByte

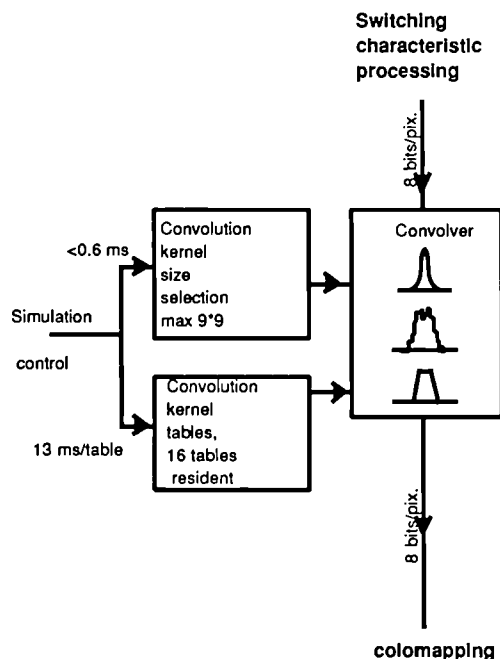


Fig. 5 Point spread function processing

The availability of more than one table during simulations is important for head movement simulations in which the perceived switching characteristic might be different for various viewing angles. The architecture of this part gives the possibility to change the 64 kByte SRAMS, in which the tables are stored, for 128 kByte SRAMS as soon as they are available. This gives the opportunity to download 2 times the amount of tables as mentioned in advance of simulation. The throughput will be 9 ns/pixel. In the case of greylevel/color simulations it will also be possible to bypass this part of the machine by loading a transparent table (64kByte).The configurable table has to be downloaded in advance of simulation, this takes about 10 secs for a complete configurable table. Selection of one out of the available tables will be possible during frame retrace.

2.6 PSF processing

Figure 5 explains the functional architecture of the Point Spread Function processing part.[6]

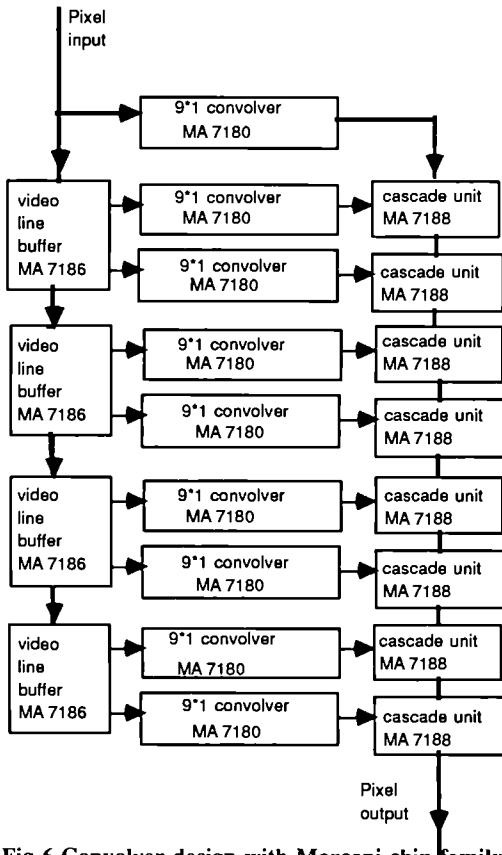


Fig 6 Convolver design with Marconi chip family

As explained in figure 6 the convolver itself is built up from a family of Marconi components. Input is coming with 8 bits/pixel from the switching characteristic processing part. The convolver calculates the influence of neighbouring pixels (max. 9*9) on a particular pixel. This will be done in real time with a throughput of 9 nsec/pixel. In this part about 20 Giga operations per second are needed. That implies the necessity to use eight parallel processes in this part. Convolution kernels can be loaded in advance via the simulation control bus, coefficients for 16 tables can be stored in the part convolution kernel tables (which could be the microprocessor SRAM for instance). Using the stage holding register of the convolver chips switching between the tables can be done during frame retrace, within 0.7 msec. Loading of the tables during simulation, so filling new coefficient for the convolution kernel in one of the 16 tables during simulation, will take about 13 msec/table.

2.7 Colormapping; combining reflected and generated image

Colormapping is a quite easy to understand process, basically this is a conversion table. A downloadable memory with as much addresses as the input to this table can generate. Two images form input to the table; the generated and the reflected image. Both provide an eight bit input this means that 64k addresses of 24 bits are available in the table. The two images are combined (added) and translated to one out of 16 million colors. Due to the delay in the switching characteristic and PSF processing part the reflected image has to be delayed (that will be done by retarding the synchronisation signals which are used to read the reflected image). Figure 7 explains the color mapping process. The table is only loadable in advance of the simulation, this will take 30 sec. For the time being only one table (192 kByte) is foreseen, as it is quite clear that during headmovement experiments also the color coordinates of displays change we have to investigate if and how we can implement, in an intelligent way, selectable tables for head movement experiments in this part of the machine, without to much increase of memory costs.

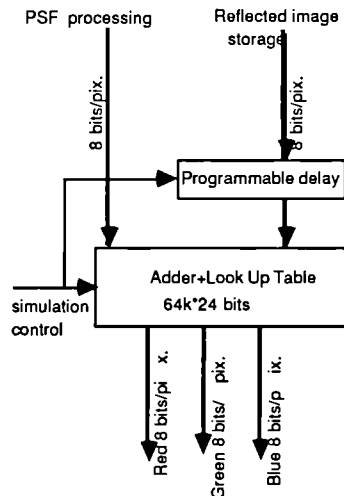


Fig. 7 Colormapping; adding reflected and generated image

3.0 Organic architecture

The total system will be split into three different 19 inch cabinets, mounted in a 19 inch rack, with in addition the Barco Industries monitor. Figure 7 shows the architecture in its main components. Four major components have to be considered

- DLX: simulation control.
- Triade 60: image generation, image storage
- Dedicated high speed hardware: switching characteristic processing, PSF processing, color mapping
- Display unit: visualisation of simulated display

Most of these parts are developed and built up by different partners and it has to be ensured in an early phase of the project that we avoid as much as possible integration problems during the later project phases. From the hardware point of view five interface connections have to be considered, all of them indicated and numbered in figure 8.

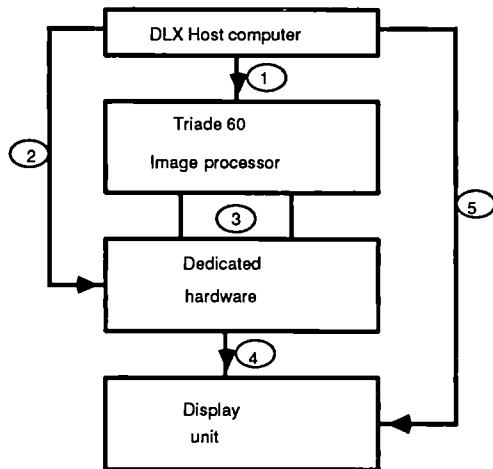


Fig. 8 Organic architecture

4 SOFTWARE SPECIFICATIONS AND INTERFACES

4.1 Introduction

The purpose of this chapter is to provide an identification of the interfacing locations, followed by the functional description of the software needed in the cover layer to be able to use and control the hardware devices. This is in fact a high level description of the interfaces. The objective is to have enough software covering all hardware devices to allow a high level software interfacing with the software simulator.[7]

4.2 Definition of low end cover layer interfaces

From the operator of the simulator point of view, the software, apart from containing models of the different technologies of FPD, has to allow the control and initialization of a simulation session, to send data to the hardware and besides it has to be able to control this hardware.

At this point it is needed to have the software piece more near the hardware (low end cover layer) performing certain functions to enable the operator to handle the system. Basically these functions can be contained in three main groups:

- loading tables
- control of hardware + monitor
- bitmap manipulation.

To be able to arrive to a well detailed definition it is previously needed to have a clear picture of the physical characteristics of the devices, their performing capabilities and their interconnections. Some of these points have already been clarified but there are still some to be considered.

The total system can be decomposed attending to their hardware components in:

- the TRIADE 60 of CIMSA-SINTRA
- the dedicated Myfra hardware
- the Barco Ind. monitor

These interfaces with the DLX have been described by three different aspects:

- its hardware or physical interface
- the UNIX/DLX interface (drivers)
- the software interface

The interfaces and function calls are well defined in the workingdocument of the project's workgroup 2, this document (Océ ref. 8623.106) is available on request.

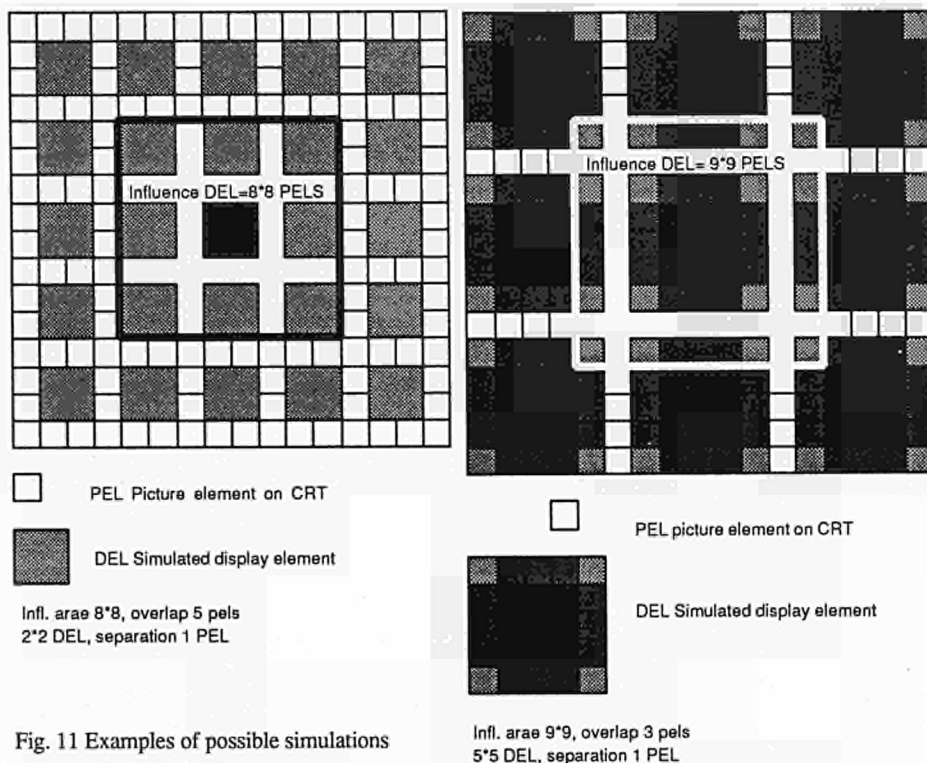


Fig. 11 Examples of possible simulations

5.2.2 Monochrome monitor

Screen diagonal: 38 cm
 Line width: 0.11 to 0.15 mm
 Scan non linearity: 1% of picture height.

5.2.3 Color monitor

Screen diagonal: 16 or 20 inch
 Spot size hor.: 0.28 mm
 Spot size vert.: 0.35 mm
 Pitch: 0.21 or 0.26 mm
 Scan non linearity: 1% of picture height.

5.3 Temporal properties

5.3.1 Scanformat monitor

Number of visible lines: 905
 Visible pixels/line: 1280
 Unvisible pixels/line: 64
 Pixel time: 9 nsec.
 Line time: 12.096 μ sec.
 Line frequency: 82.67 kHz.
 Frame retrace time: 1.0 msec.
 Frame frequency: 83.7 Hz

5.3.2 Switching charact. simulations

The simulator operates with a frame frequency of 83.7 Hz. The hardware is capable to calculate for every PEL every 11.9 msec. a new value for the luminance during on and off switching of a simulated display element.

Performance:
 luminance domain: value represented by 8 bits (256 levels)
 time domain: samples every 11.9 msec.

5.3.3 Monochrome monitor

phosphor decay time: in 10 msec to 2%

5.3.4 Color monitor

phosphor decay time: R 8 msec to 2%
 G .8 msec to 2%
 B .35 msec to 2%

6 CONCLUSIONS

Success of the project will accompany the introduction of a young technology, being flat panel displays, with an user driven approach, rather than a technology driven approach. This can help to avoid the mistakes made during the introduction of the first generation of Visual Display Units. Of course this does not decrease the relevance of the simulator machine for technology development.

After the end of the project this simulator facility will be available for interested European parties, such as display manufacturers, display applicators, ergonomists, etc.

As duplicating of the facility is rather expensive, a study group has been installed to investigate the feasibility of porting a subset of DISSIM on a standard engineering workstation making DISSIM available as a desktop product. During ETW 1988 we hope to be in a position to report about the forementioned possibilities.

ACKNOWLEDGEMENTS

The author would like to express his appreciation for the cooperation of all projectmembers, the members of DISSIM workgroup 2 for the usefull discussions during various workshops and especially I. Placencia Porrero, for her helpfull and encouraging assistance during the project .

REFERENCES

- [1] Bosman et al, ETW 1985, Modelling and simulation of the visual characteristics of modern display technologies under office work conditions (Nort-Holland, Amsterdam 1985)
- [2] A.É. v. d. Meulen, Technical Annex OS 1593, Océ Ned. B. V. ref. AVDM/CAB/8623.067
- [3] M. Versteeg, Display simulator; functional architecture, TUT ref. 065.1495.
- [4] I. Placencia Porrero, Concept of the software structure for the Esprit project OS 612, Océ-Ned. B.V. ref. IP/DRI/8522.151
- [5] J.P. Levis, Report top level design convolver, Myfra.
- [6] J.P. Levis Final report: Luminance Controller and Color Look Up Table, Myfra.
- [7] I. Placencia Porrero, A specification of the main modules used in DISSIM Software concept, Océ-Ned. B.V. IP/SVO/DRI/8622.080
- [8] N. Fielibert, Report on CRT specifications OS 612 + extension, Barco Ind. ref. NF100486JB/2099

Project No. 295

THE ROLE OF PAPER IN THE AUTOMATED OFFICE

U.Boes *, *W.Doster* §, *G.Fogaroli* °, *H.Löbl* ‡, *G.Maslin* €

* AEG Aktiengesellschaft, Konstanz, Germany

§ AEG Aktiengesellschaft, Research Center, Ulm, Germany

° Olivetti DRT, Ivrea, Italy

‡ Philips Research Laboratory, Hamburg, Germany

€ Plessey Network & Office Systems Ltd., Beeston, United Kingdom

Abstract

This paper discusses the role of paper in the offices of today and of tomorrow. The need for establishing a link between the paper world and the electronic world in both directions is elaborated. The following aspects of ESPRIT 295 project dealing with the paper interfaces are considered: FROM, WITH, TO PAPER subsystems and their integration, especially, human interface, system interface and potential applications.

1. INTRODUCTION

Paper documents and the electronic representation of information, together will form the backbone for the office of the future. Each of the representations has its own specific advantages and disadvantages. To profit from the advantages it is necessary to bridge the gap between these two representations without losing any informational contents. This demands powerful tools for the automated transfer of information between paper-based and electronic-based documents and vice versa.

ESPRIT Project 295 "The Paper Interface" aims to provide these tools. To reach this goal the project is focussed on the four main aspects FROM, WITH, TO Paper and INTEGRATION.

In the FROM PAPER part, composed multi colour paper documents are automatically scanned and analysed. Handwritten and printed text as well as line graphics are recognised and transformed into their electronic equivalent.

WITH PAPER deals with the on-line recognition of handwriting and line graphics in real time which means recognition during the writing process. The results of the recognition process in the FROM and WITH part are improved by advanced post processing techniques and transformed into a standardised data format.

The TO PAPER part transforms these standardised data into the representation necessary for displaying and printing and provides an advanced colour printing technology.

The FROM-WITH-TO subsystems are integrated on top of a state-of-the-art workstation. The means for the generation and handling of documents by editing, formatting and imaging processes are provided in a common system having in mind the requirements of a modern human interface.

Within this INTEGRATION aspect there are the following main activities: the human interface, internal data representation, a system interface, and the demonstration activity. Solutions shall not only exist on paper or within documents but shall be implemented and demonstrated in practice.

This paper deals with the role of paper in the automated office, considers the FROM-WITH-TO subsystems and then sketches the system and human interfaces and potential applications.

2. PAPER IN THE OFFICE

Paper has played a central role in offices for many years and it will play a very important role in the offices of the future. Why is this? To give an answer to this question we will consider the advantages and disadvantages of paper and paper-based offices on one hand and of electronic-based offices on the other hand.

Handling of documents, basic requirements, qualifications and the human interface

Usually people are very familiar with paper. Everybody is able to handle paper as a carrier of information. Paper itself has a rather good human interface. The introduction of computer systems for document handling is ongoing. Yet their user acceptance is limited due to difficult handling and problems like screen resolution, illumination of the room and flicker. To operate such a system additional handling is required when compared with paper handling.

Creating and Editing Documents

Everybody knows how to produce documents with a variety of different fonts, tables, figures and even photos. Creating mixed-mode documents therefore seems to be rather easy.

Editing of paper documents to some extent is also easy and possible e.g. using rubbers, fluid correction materials, and cut and paste technique. If the modification degree increases, the ease of correction decreases and also the quality of the resulting paper documents. If documents are produced with any type of printing machine then additional qualification is necessary.

Creating electronic documents requires the knowledge of an editor. Advanced available systems allow the production of mixed-mode documents, but most editors only allow the handling of text. Changing those documents is really easy without influencing the quality of the final document.

Storage and retrieval

To store information on paper no external energy is necessary, and no electrical or magnetic influence can destroy the information. Stored in filing cabinets in an office, paper documents can survive for many decades. Problems can, however, occur with humidity or fire. The access to private filing cabinets is rather easy, people remember the file and quickly find the required paper document.

If organisations grow an additional problem occurs - the mass problem. The mass of paper always requires for a lot of space. Retrieval within such voluminous paper stocks often means heavy work and usually implies a rather high access time.

Storing electronic documents on disks, tapes, etc. allows more flexibility in accessing them. In principle even an associative access is possible. Very powerful and quick retrieval algorithms can be applied.

Transfer and distribution of documents

For paper documents there is no need for an explicit document exchange format; there are some implicit formats, memos e.g. usually have a different appearance from bills. But even if this implicitly agreed format is not used, it is possible to exchange paper documents and to have access to the contents. In the area of photocopiers it is also very easy to distribute even a lot of paper documents by simply copying the original document.

Electronic documents are usually transferred by three different methods: On storage media e.g. floppy, as electronic mail or as printout. Electronic document transfer can only be managed easily if sender and receiver use the same editing program or sometimes even the same hardware. For general electronic distribution, document interchange formats and document architectures have to be defined and applied, e.g. TELETEX, ODA/ODIF /1/.

Trends

Office automation is continuously emerging from concepts to realisation. Within organisations the electronic office will become standard at least for organisation wide communication. Also smaller companies will install their PCs and will also do some office automation.

On the other hand paper will remain a standard medium for information transfer. New developments in desktop publishing /2/ and laser printers force the intensity of using paper /3/. People want to produce high quality documents; today it is only possible to do it via paper.

Even within fully computerised organisations paper is important as long as there doesn't exist an equivalent electronic medium that can be handled by unexperienced users. People will print, read, correct, modify and store paper documents themselves.

All these aspects clearly define one demand: Devices are necessary to realise a one-to-one mapping from paper to electronic representation of documents and vice versa. Project 295 is focussed on the design and development of these devices which for the user will combine the advantages of both media.

3. FROM PAPER

This task can be described in one sentence: Transforming paper documents into their equivalent electronic representation for further processing by editing etc. Activities are focussed on

- Multicolour Scanner Development
- Image Analysis
- Analysis of Graphics
- Final Demonstrator
- Handwriting Recogniser.

The general approach is described in /4/. In this paper we will describe work being done in the area of multicolour scanner development, organisational aspects of the final demonstrator with respect to image analysis, and in the analysis of graphics.

Multicolour scanner development

Besides electronic transfer of scanned images the scanner, in general, can be considered as the input link between the office computer and the outside paper world. Paper information is of two kinds: information which can be read, interpreted and coded, and information in bit map mode (pictures, in general), which has to be somehow reproduced in the best possible way. For the first type high resolution, good contrast, capability to separate information from background, flexibility in handling document sizes are required. For the second type, mainly gray level and colour detection capabilities are required.

Moreover, speed and sometimes automatic document feed are also very important in both cases. The multicolour scanner being designed in this project has all the functionalities necessary to comply with the above described requirements: these functionalities have been implemented through a number of advanced solutions as:

- Specifically designed optical system with high efficiency and MTF objective lenses plus colour separation prisms integrated with dichroic filters.
- Automatic high sensitivity luminance threshold in order to enhance contrast, resolution and background independency when binary mode, for OCR recognition, is used.
- Sophisticated compensation and conversion devices to acquire the highest number of RGB primary colour levels.

The most interesting features of the scanner design are:

- Resolution up to 400 dpi.
- Document size up to A3 format.
- Flat bed or pass-through with automatic document feeder.
- Binary or multilevel scanning modes (monochrome or colour), depending on the selected function.

A typical operation would consist of scanning at high speed, black and white binary mode, and with a resolution consistent with the finest details that have to be detected. At the same time a colour indication will point out the coloured areas of the documents. When a picture (bit map defined) area has been detected, a second scan, limited to that area will be performed; this time all the information content, be grey or full colour, will be preserved, to be subsequently processed (e.g. compressed) according to the selected use.

A special care has been dedicated to document handling mechanisms and to ergonomic aspects of the prototype implementation. A guideline for the scanner design has also been the objective of deriving possible low cost future products.

Organisational aspects of the final demonstrator with respect to image analysis

The recognition of documents cannot be done in a straightforward manner. This means that the recognition system must be able to go back to previous intermediate results (backtracking). Therefore the activation of the single recognition algorithms must be very flexible and dependent on input data. The intermediate results of the algorithms have to be structured and stored in a useful manner. Fig. 3.1 shows the principal organisation.

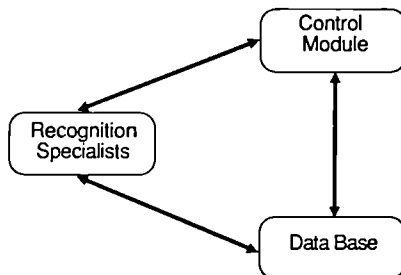


Fig. 3.1: Principal organisation within the final demonstrator.

The control module supervises the recognition specialists and conducts the whole process. This allows to deviate from the sequential processing order: feedback is rendered possible. The recognition specialists get their input data from and store their output into data bases. This data structure includes the textual, graphical and pictorial components in a hierarchical order. The Reader Specific Data Structure (RSDS) will be implemented as an abstract data type to gain independence from the actual implementation.

In the current status of the system the recognition specialists are the detector of photographic images, the connectivity analysis and the text/graphics-categoriser. Work is continuing to implement methods for recognition of textual and graphical elements. The current status of the implementation of the image analyser is demonstrated in fig. 3.2 (a) through (c).

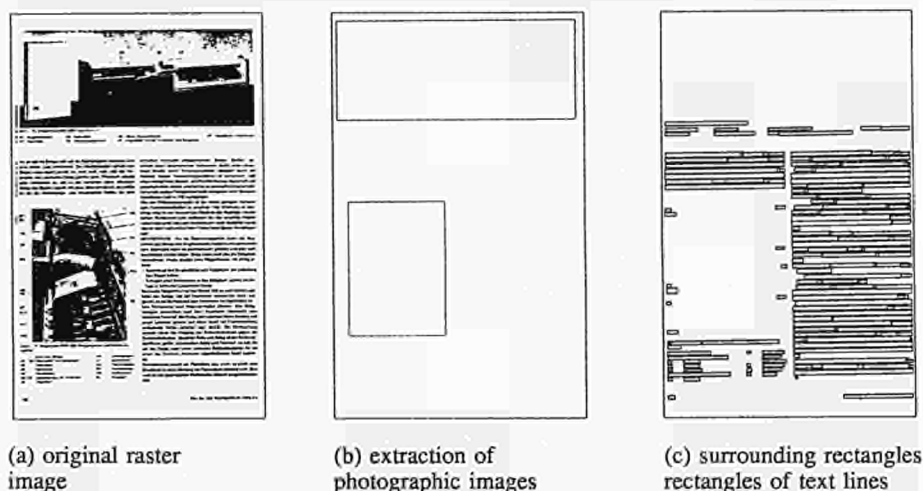


Fig. 3.2: Sequence of images demonstrating the current implementation status

Analysis of graphics

After the scanning process a raster image of the page is the basis for further processing. A subsequent connectivity analysis results in the database containing all connected regions together with their border lines, area measures of roundness, etc. and a description of the hierarchical relationship of the objects. Text/graphics discrimination -which is not considered here- yields potential text and graphics objects which then are further processed, text is recognised and graphics objects are analysed. In the following some aspects during the process of analysing graphics objects are considered.

Analysing of graphics objects means more than a simple vectorisation of lines. Knowledge about the application context is necessary and is used to handle the problem. Fig. 3.3 shows some typical graphics objects which occur in office documents and which have to be analysed.

All analysis consist of two parts, analysis of allowed symbols within the application context and if necessary analysis of connecting lines. The symbols in fig. 3.3 consist of line graphics, which are arranged to form two-dimensional symbols. Assuming that the surrounding lines are connected, which can be achieved by suitable preprocessing operations, a symbol always has an inner and outer contour and may be considered as a superposition of a white area on a slightly larger black area.

In the case of flow charts the potential symbols can be mapped to circles, rhombi, rectangles or ovals, in the case of pie charts to pies and the case of diagrams to rectangles. Applying simple measures for description of dimensions and shapes makes it possible to classify the potential symbols. A certain similarity among some symbols requires the application of additional features as height/width ratio and focal radius ratio. Fig. 3.3 illustrates also the extracted symbols of the graphics objects. Comparing the analysis of graphics with the recognition of text we can think of the symbols to be characters of a word and the connecting lines to represent the information how to arrange characters to words.

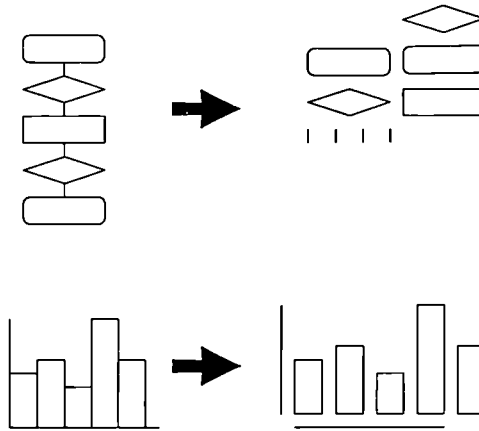


Fig. 3.3: Examples of graphics objects and the extracted symbols of these objects

Analysis of the connecting structure is based on the line approximated representation of the graphics. Lines which are part of classified symbols are labelled and extracted. The next step is to combine lines running into the same direction within a given tolerance and having closely neighbored starting and ending points to longer lines.

Then parallel and duplicate lines are reduced to central straight lines. Supplemented by syntactic analysis oriented procedures it is possible to handle graphics structures of the described complexity.

4. WITH PAPER

Script recognition

On-line script recognition of handwriting and line graphics in realtime during the writing process on a graphics tablet is the topic of the WITH PAPER activities.

The objectives of the script recognition work are ultimately aimed towards electronic paper, see fig. 4.1, allowing the user a natural medium for generating, editing and annotating documents on-line. The work is broken down into three key areas:

- recognition of the lower case alphabet, initially with a constrained user base.
- progression towards a user independent lower case recognition system, and the extension of the character set to encompass upper case characters and numerals and some special characters. Examination of some basic natural editing functions.

- investigation into cursive handwriting and migration of the non-cursive algorithms to cursive handwriting. The initial aim is a writer dependent system. Investigations towards writer independence will be assessed.

This work is described in more detail in the paper of Philip Wright /5/.



Fig. 4.1: Future "electronic paper" system

Graphics recognition

The objective of dynamic sketch recognition is to render a hand drawn sketch to a drawing of draughtsman-like quality, in real time. That is, as a person draws a sketch he will see each stroke neaten-up and geometric objects adjusted for alignment and regularity. It is intended for use in interactive drawing composition and to be completely user independent. It is anticipated that pen driven editing will be used in conjunction with sketch recognition so that the recognised drawing can be corrected or amended in a highly interactive way.

We use the word sketch to mean the kind of drawing most likely to be found in an office document, e.g. pie chart, histogram, tabular form, isometric diagram etc, in a style which is non artistically biased i.e. no shading, just outline drawings. We hence define a sketch to be a series of hand-drawn lines intended to represent geometric objects which may compose higher level contextually meaningful objects. Some examples are shown in figure 4.2.

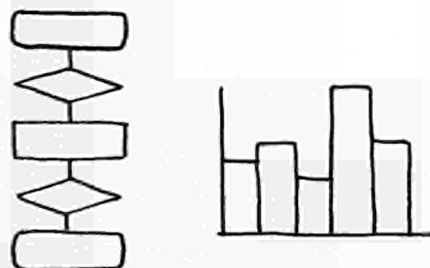


Fig. 4.2: Targets for sample sketches

The recognition process is performed dynamically and hierarchically. That is to say, as the user draws, lines and arcs will be recognised and assembled, and while this assembly is taking place basic shapes will become discernible and connect into higher level objects. The displayed rendition of the drawing is therefore improved at each level of recognition and in this way the user always draws in relation to the recognition so far. The process of structural recognition can be considered as a series of discrete phases shown in figure 4.3 in increasing order of abstraction from the original hand-drawn input.

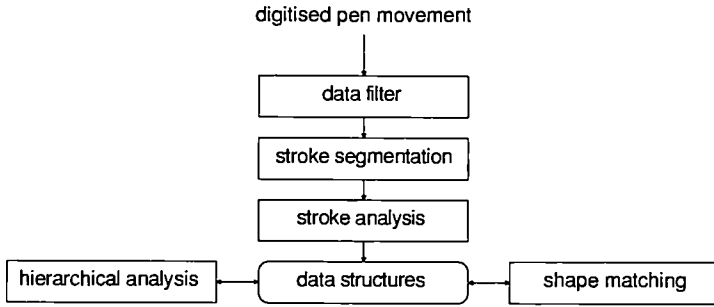


Fig. 4.3: Structural pattern recognition process

Data structure:

A model of the recognised sketch which evolves as drawing proceeds. It is manipulated by the surrounding recognition phases which maintain a hierarchical structure of data elements, viz: a complex shape recognised by hierarchical analysis is composed of shape elements and line elements. Shape elements recognised by shape matching are themselves composed of line elements which are recognised by stroke segmentation and modified by stroke analysis.

Stroke segmentation:

Monitors the movement of the pen and outputs geometric lines and arcs in the first step of neatening the drawing.

Stroke analysis:

Performs modifications based on heuristic assessments of the user's intentions, such as merging together parts of a line which have been separately drawn. These fitted lines and arcs form the basis of the sketch data structure.

Shape matching:

Monitors the assembly of lines and arcs, and enhances the sketch data structure with elementary shapes.

Hierarchical analysis:

Recognises higher level objects that evolve out of shapes and lines.

To date, stroke segmentation and analysis have been implemented, and after a review of the literature a technique has been selected which will form the basis for shape matching. It represents an improvement over traditional relation correlation techniques in that it employs global geometric relationships between a shape's component lines, as well as local connectivity features. Traditional methods have been limited to local relations and are consequently more sensitive to local deformations. The selected technique is, however, very sensitive to the order in which a shape's component lines are drawn because of the way local connectivity is characterised, and because sequence numbering is used to qualify global relations. This is totally unsuited to the problem in which component lines are drawn in an unpredictable sequence, and these attributes will be removed from the technique and other suitable characterisations will be devised. Some analysis of the line structure is needed as a preliminary to shape matching. For each line, connections and relationships with other lines in the drawing need to be established. This has been retrofitted into stroke analysis to be done on a latest line basis.

5. TO PAPER

The increasing complexity of office documents, which may contain text, raster graphics and geometric graphics together on the same page requires high performance printing devices. A further step towards new office applications will be the addition of coloured business graphics and even multi-colour images to office documents.

Electrophotographic printers have been constantly improved over the last years with respect to print uniformity, resolution, printing speed and printing costs, so that the generation of high quality monochrome office documents is basically possible. The question is whether the high standard of performance given with monochrome laser printing can also be achieved in colour printing. Electrophotographic colour printers could probably cope with most of the office requirements, but the printer costs will be too high to employ such a printer on a workstation.

Because of the variety of available printing devices many different incompatible document formats are in use for these printers. It is likely that this multitude of vendor specific printing formats will disappear due to emerging de facto standards or international standards.

The TO PAPER activities aim towards the following objectives:

- Assembly of a printer controller and implementation of available de facto standards or international standards suitable as printer document format.
- Assembly of high resolution printing devices for monochrome and colour printing and connection of these printers to the printer controller.

Fig. 5.1 shows the whole hard copy rendition unit as it should be available at the termination of the project. A printer controller which is connected to the workstation via a SCSI-Bus supports a monochrome and a colour printing engine. The printer controller is currently emulated by a VME-based general purpose computer with a 68020 CPU.

Printing Devices

At present two monochrome printing engines have been employed, an electrophotographic device with a magneto-optic light switching array and a laser printer. These devices print with a maximum speed of 12 pages/min and 15 pages/min respectively, both at a raster resolution of 12 dots/mm.

For colour printing the ink jet technology was chosen, since this technology is basically suitable for low cost printers with low cost consumables and high print quality. The printing speed and raster resolution should come close to those of a thermal transfer line printer. It was further decided to employ a drum-type device with a special print head that is currently being developed. This print head is shown in fig. 5.2. It is based on a new microplanar technology /6/, which allows for high integration, small dimensions and high droplet rates of up to 10 kHz per nozzle.

The colour printing engine will have the following features: a 24-nozzle linear print head with 6 nozzles each for yellow, magenta, cyan and black inks, variable raster resolution between 8 and 12 dots/mm and a printing speed of about two minutes per A4-sized document at a raster resolution of 8 dots/mm.

The microplanar technology allows further improvements beyond the above mentioned performance, i.e. an integration of more print elements, still smaller dimensions and the use of melt-able inks for printing on rough uncoated paper.

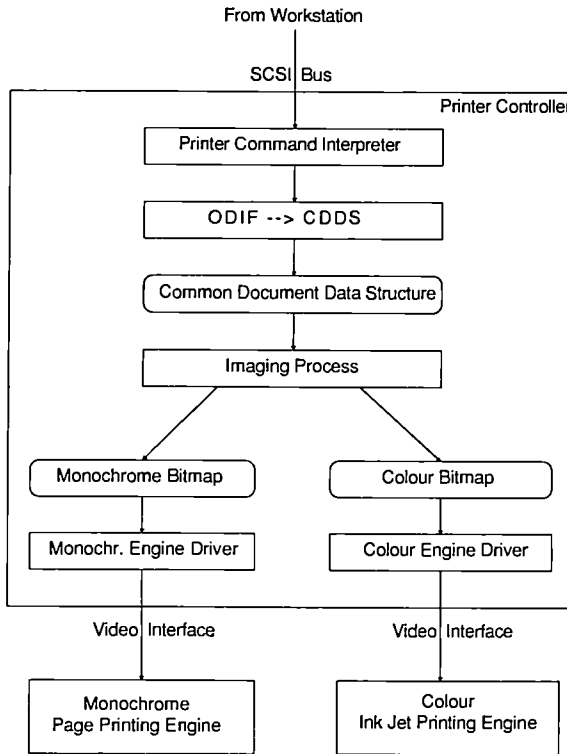


Fig. 5.1: Hard copy rendition unit with ODIF printer controller

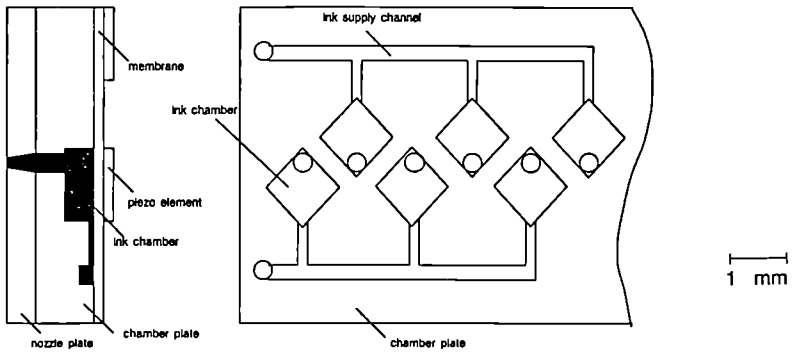


Fig. 5.2: Microplanar multi-nozzle ink jet print head

Printer Document Format

The simplest document format that can be transmitted to a printer controller is a raster file. However, it is desirable to perform the conversion from a printer independent document representation to a printer dependent bitmap form by the printer controller. It was decided to use as such a device independent document format an internationally standardized document interchange format, the Office Document Interchange Format (ODIF) from ISO DIS 8613 /1/ and test its suitability as printer document format. A first implementation for monochrome documents with test and raster graphics should be available rendition of documents with geometric graphics and multi-colour images.

Another approach uses PostScript, a page description language /7/ which is currently being established as a de facto standard for a printer independent document description. Due to the increasing importance of PostScript, a transformation program is being developed that is able to convert the workstation internal representation of a document into a PostScript file, which can be sent to a commercially available monochrome PostScript printer. As in the case of ODIF transfer only documents with text and raster graphics can be handled at present, but the extension of geometric graphics is in preparation.

6. INTEGRATION AND APPLICATIONS

System aspects deal with a variety of important objectives. These objectives include: an integration of the different sub-systems, a common internal document data structure that permits an optimised exchange of information between these sub-systems, the selection and use of a standardised office document architecture and document interchange format, the generation and editing of complex documents and finally the design of a suitable human interface.

A completely satisfactory solution to all these requirements cannot be achieved within the framework of the PAPER INTERFACE project. However, it is intended to prepare a final demonstrator that could be used as the basis for future office products.

In order to permit the handling of office documents by the sub-systems and the workstation, suitable document representations and interchange formats must be employed. A central document data structure was defined, which serves as a common interface between the sub-systems. The software modules interact via such a data structure, which was termed Common Document Data Structure (CDDS). It is based on a standardised office document architecture /8/, which was selected to avoid that results might become incompatible with future office requirements.

An appropriate solution is offered by the office document architecture model from ISO DIS 8613 /1/. This model provides a set of rules for an Office Document Architecture (ODA). It allows for logically and layout structured documents consisting of contents text, raster graphics and geometric graphics, which are specified by so-called content architectures. These content architectures will be extended in the framework of the project by new content architectures for grey-scale and colour images.

ISO DIS 8613 specifies also a document interchange format, known as Office Document Interchange Format (ODIF), which will be employed for communication purposes between host systems and for the storage of documents. Although the structure of an ODIF file is not suitable for direct manipulation on a document, like editing or imaging, it will be employed for the transmission of document data from the sub-system to the workstation and vice versa.

The corresponding conversion modules will probably be used in the near future for products that allow the interchange of documents between workstations of different vendors, which is

still rather difficult, given the incompatible software and document data structures used by most of these vendors.

Manipulation on the document data originating from the document reading unit or the tablet recognition unit so-called Reader Specific Data Structures (RSDS) and Tablet Specific Data Structures (TSDS), will be performed after their conversion to the CDDS. The document manipulation system which will consist of three modules, an editing, a formatting and an imaging module act all on the CDDS. At present, however, only the imaging module is in operation. This module is able to perform a conversion from the CDDS of a document with text and raster graphics content into a bitmap for rendition on a screen. Imaging of geometric graphics, grey-scale and colour images is in preparation.

The formatting and editing modules will be implemented in an object-oriented way by using the programming language C++ /9/, which is an object-oriented extension to C. Object-oriented programming allows the definition of object classes which represent exactly the logical objects and the layout objects of ODA documents. The formatting and editing modules will not be completely implemented within the framework of this project. It is primarily intended to demonstrate the integration of the sub-systems and their functionalities and to prove that the use of ODA can lead to attractive document handling features that are appropriate for the needs of future office systems.

Human interface

Human interface in this context is the interface between person and machine. In order to enable the user to work efficiently with increasingly powerful machines, a powerful interface has to be designed with respect to the properties of the user /10/. Humans frequently think in images, therefore visual symbols are often more informative than text. Thus, in current PC or workstation design there is a trend towards graphical user interfaces for operating computers /11/.

As existing methods are inadequate for document analysis new methods of presentation have to be developed. Besides visual representation of all types of information the information input via keyboard, colour scanners, tablet and mouse systems is also part of the human interface.

In the field of document analysis it is essential to scan a document, display it on a screen, modify and store it. A document can consist of an arbitrary combination of photos, graphic and text. A recognition algorithm deals with distinguishing and encoding these components. The representation of all document components in one image is important for the arrangement of the document. At the same time there must be the possibility to select any part of the document for editing purposes.

For text correction an efficient environment is necessary including a facility for interactively selecting the correct character from different recognition alternatives. For this purpose for example the display of these alternatives in different colours would be helpful. Fig. 6.1 shows a potential arrangement for displaying and selecting alternatives.

Human interface is also useful for the development of recognition algorithms since the presentation of intermediate results provides efficient test facilities. The recognition algorithms can then be divided into functional modules and optimised separately. An example for an adequate environment for the development is shown in fig. 6.2.

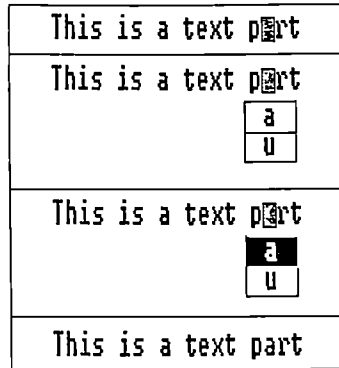


Fig. 6.1: From top to bottom line: original presentation on the screen - Opening the alternatives' window - Selecting one alternative - Final result after substituting

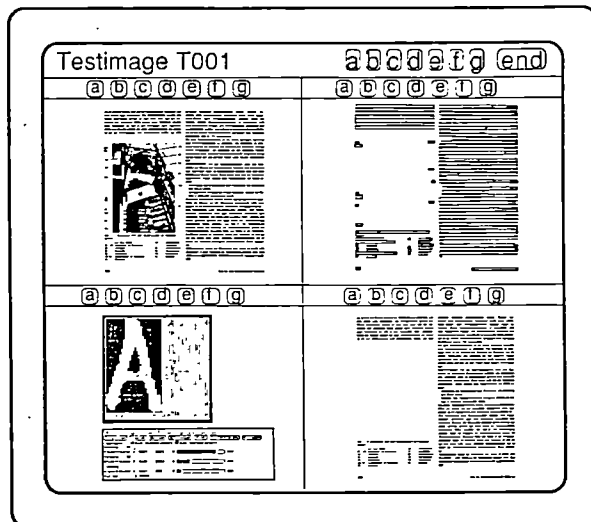


Fig. 6.2: Presentation environment for the development of recognition systems

The operating system used in the demonstrator environment -a SUN-Workstation- is UNIX 4.2 BSD, a multiuser, multitasking operating system, which is well suited for a multiwindow high performance user interface. With respect to the user interface itself the already existing layers of the SUN user toolkit will be employed as a basis for our future work.

Besides the development tools as described in this chapter, the most important tools which must be implemented are tools for the interactive elimination of character ambiguities from the character recognition process and an integrated document editor consisting of document generation modules, imaging modules, and formatting modules. Whether the tool for resolving ambiguities will be incorporated in the document editor or not will be decided at a later date. The document editor itself will be an editor with reduced functionalities. In the following two potential applications will be described, where the advantage of the PAPER INTERFACE easily can be recognised. These applications shall be possible with the planned final demonstrator within this project.

Preparation of Advertising Materials

The preparation of the advertising material is a very good example. The format of such documents will vary from country to country, from area to area and from target group to target group. Nevertheless, the data will essentially be the same, namely a photograph (or sketch), a verbal description of the product itself, tables, charts, etc.

The standard heading, logo etc. used by the company would be drawn from disk, as also would any standard text passages that would frequently be required, including standard phrases. Other items could be drawn from a database system on disk, such as the system data of the product.

The positions and formats of the photograph and text would be manipulated using the windowing and editing features of the workstation. A draft of a chart and hand- or machine printed text would be scanned and recognised via scanner and FROM PAPER module. Additional text would be entered and the correction of the chart would be carried out using the data tablet, the freehand script being recognised on a character by character basis and then post processed by the AI system to resolve ambiguities. Any remaining ambiguities would then be resolved by hand, using the editing capabilities of the tablet system.

Final editing of the layout and the correction of any remaining errors would then take place using the most appropriate tool for the application, and the whole would then be output on the multicolour printer. Fig. 6.3 shows a schematic representation of the preparation of an advertisement.

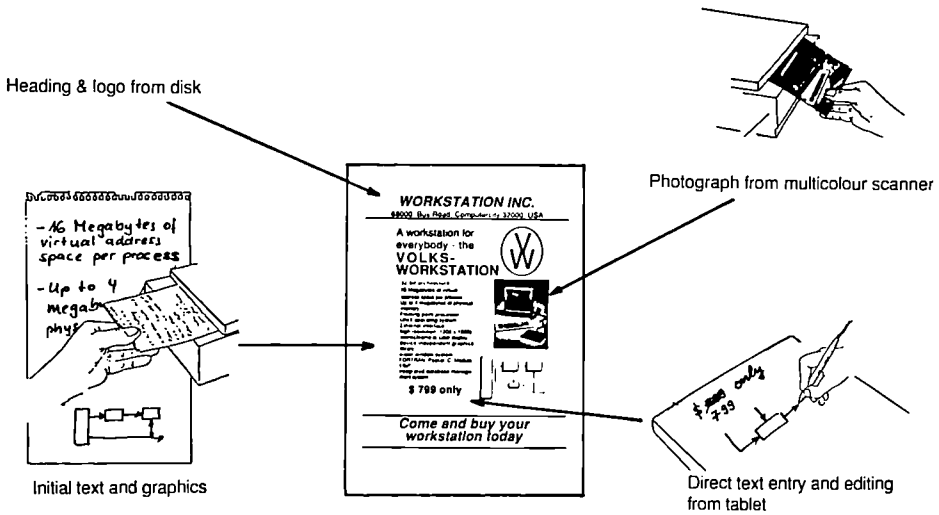


Fig. 6.3: Schematic representation of the preparation of an advertisement

Selective Reading

Selective reading is a means for document generation and preparation. One or more parts of a scanned or already electronically processed document are to be selected by user interaction. The user must select the desired parts by applying the existing user interface, e.g. by means of a surrounding rectangle. The purpose of selective reading is to reduce information for further processing steps by discarding information which is unimportant for the application. Thus, selective reading might be applied everywhere during electronic document processing, in the bitmap-representation as well as in recognised text or graphics parts. The selected part of the document must be amenable to manipulation, as zooming and pixel editing or viewing of recognition alternatives and correction of results.

Plenty of applications for selective reading may be conceived. In the context of the project 295 it is intended to be a means for document generation and preparation, as well as monitoring the recognition. A further application may be interactive abstracting with the aim of establishing a personal user data base as depicted in here, words and groups of words are framed and marked. The totality of these frames can be used as key elements like TITLE and ORIGIN for the generation of a document's profile according to ODA/ODIF, ISO DIS 8613, or for document archiving and referencing in general. Selective reading might also become a valuable help in processing and filling in forms. Each form can be described by a hierarchical tree structure according to ODA, where the logical and layout parts are represented separately. Both parts point to the content portions. The hierarchical description may be accomplished interactively, which then is the main precondition for selective reading or acquisition of filled-in forms.

Fig. 6.4 visualises selective reading applied to the processing of forms like bills or bank transfer forms. In these cases the same information types like DATE, NAME and AMOUNT are represented in different layout structures and have to be mapped into one standardised layout for further processing.

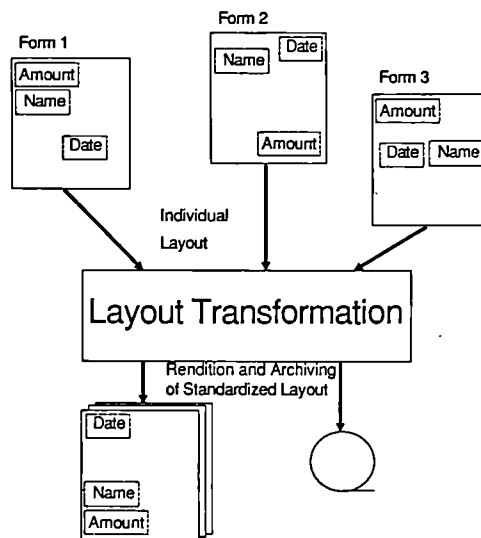


Fig. 6.4: Processing of forms with different layout and identical logical structure

7. CONCLUSIONS

Within this contribution the importance and future place of paper in the office environment is shown. This illustrates the requirement to bridge the gap between electronic and paper media by the transmission and storage of naturally created documents.

Work is continuing to further refine the FROM, WITH and TO PAPER subsystems and to integrate them into a demonstrator for the end of the project. In addition, a preliminary demonstrator is planned for the end of this year and it is believed that this project and its demonstrators will provide the basis for new advanced office applications.

References

- /1/ Information processing - Text and office systems -
Office Document Architecture (ODA) and Interchange Format
ISO DIS 8613, part 1-8, 1986
- /2/ B.Grout, I.Athanasopoulos, R.Kutlin
Desktop Publishing From A to Z
Berkeley, California, 1986
- /3/ R.J. Lahr
The Non-Death of Paper
Proceedings Journees d'Electronique et de Microtechnique, Lausanne, 1984
- /4/ U.Miletzki, Ch.Steigner, W.Doster, G.Fogaroli, H.Löbl, P.Moulds
Paper Interfaces for Office Systems
ESPRIT '86: Results and Achievements
Amsterdam, 1987, pp. 373-387
- /5/ P.Wright
Dynamic Handwritten Script Recognition
accepted paper for ESPRIT'87 Technical Week
- /6/ H.Bentin, M.Doering, W.Radtke, U.Rothgordt
Physical Properties of Micro-Planar Ink Drop Generators
Journal of Imaging Technology 12 (1986) 3, pp. 152-155
- /7/ Adobe Systems Incorporated
PostScript Language Reference Manual
Reading, Massachusetts, 1985
- /8/ A.Scheller
Stand der Normung im Bereich der Dokumentenverarbeitung
(ODA, SGML, T.73, ECMA-101)
DFN Report no 49, Sept. 1986, Berlin, pp. 115-132
- /9/ B.Stroustrup
The C++ Programming Language
Reading, Massachusetts, 1986
- /10/ B. Christie, ed.
Human Factors of the User-System Interface
Amsterdam, 1985
- /11/ Human Factors in Computing Systems and Graphics Interface
Proceedings CHI + GI 1987
Toronto, Canada, April 5-9, 1987

Project No. 295

DYNAMIC HANDWRITTEN SCRIPT RECOGNITION

Philip Wright

Plessey Networks and Office Systems Ltd.
Trafalgar Rd.
Beeston
Nottingham
England

This paper describes a method for the automatic recognition of handwritten lower case characters as a basis for the analysis of cursive handwriting. Characters are identified by encoding of their direction of travel which is compared to pre-analysed stored direction encodings. The construction of the encoded character databases is described, along with the technique for multi-stroke character matching.

1. INTRODUCTION

This paper describes the work undertaken so far in the analysis of dynamic handwritten script within the bounds of the Esprit Project 295 - The Paper Interface. This is one of three main areas being researched by this project.

- scanning of a pre-produced office document which may contain text, line drawings and photographic elements. Recognition of the textual parts (either typewritten or handwritten characters) and graphical elements and identification of the photographic elements.
- the dynamic translation of handwritten documents, interpreting text and graphical information, and allowing interactive document editing.
- the generation of a paper document from electronically filed information regardless of the means by which the electronic data was originally created.

The work on dynamic script recognition is covered in three stages:-

- the development of algorithms for recognition of lower case unconnected characters with some degree of user dependence.
- progression of the technique to a user independent environment through the detailed analysis of a larger number of writers. Substitution of the data set so as to observe the performance of the algorithms on the upper case data set, numerals and some special characters.
- extension to the analysis of cursive script. Feasibility as to the creation of a user independent cursive script recognition system.

It was decided to concentrate initially on unconnected lower case letters, although it is realised that a recognition system allowing only unconnected letters (upper and/or lower case) would be quite unacceptable to the vast majority of potential users. Many of the people sampled found it very difficult to write characters unconnected, especially when writing at their normal pace. Analysis of the lower case shapes produced by people writing both cursively and non-cursively showed that the vast majority of characters had very similar shapes in both cases. Only a small subset of characters was found to have different styles of formation, eg.

b, f, k, p, s, t, x, z.

The major aim of the work on unconnected letter analysis is to use it as a basis for the analysis of connected script. The two major approaches to cursive script recognition are word analysis [1], which tends to limit the vocabulary scope, and character segmentation and recognition [2]. The latter has the advantage of a far wider vocabulary but is more difficult due to the problem of character segmentation. However, the character feature analysis performed on the unconnected characters would also be suitable for the segmented characters or part characters.

A graphics data tablet was used to capture the pen tip motion while a user wrote. The data tablet used was a Numonics 2200 connected to an Olivetti M21 portable PC. Data files were downloaded onto a VAX 750/SUN network where the algorithms were developed. The algorithms were then downloaded onto an Applications Processor (a Force 68020 microcomputer) in order to assess real-time performance. Configuration of the initial demonstrator is given below:-

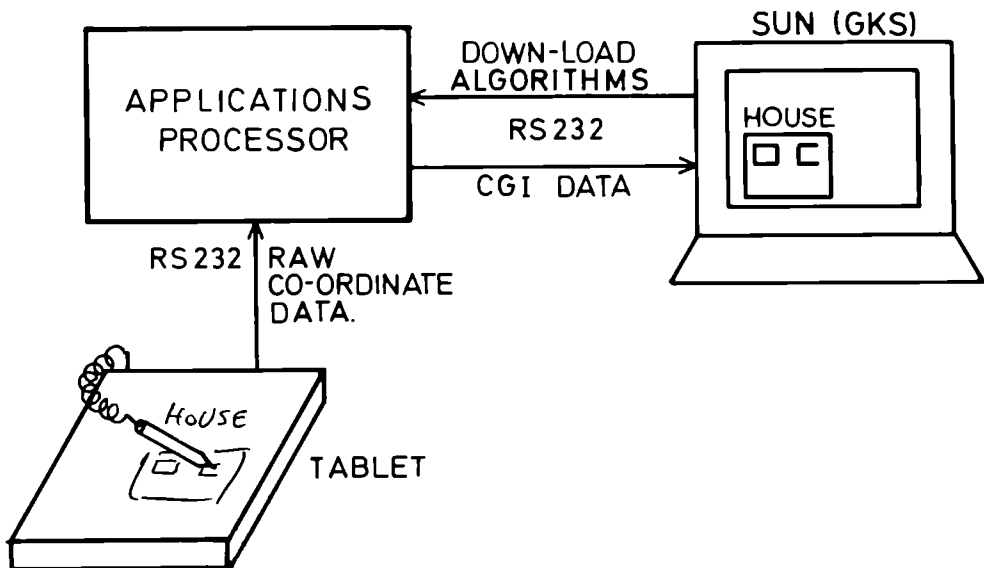


Figure 1 - System Configuration

The GKS (Graphical Kernel System) resident on the SUN, was used to display the recognised script and graphical information. The motion of the pen tip is converted into a number of directional sequences. When a pen-lift is detected, the encoded directional sequences are normalised over the character path and the character encodings compared to a previously compiled

database in order to find the closest match.

Two different techniques are used to give separate recognition results,

1. A technique which encodes the pen travel into localised changes in the x and y directions by detection of the x and y turning points.
2. A technique which encodes the character path into a number of quantised vectors, increments of the curve being approximated to one of eight possible vector directions.

Results from the separate algorithms are correlated to give the overall best-fit character identifier.

Initially, a number of preprocessing steps were investigated:-

- (i) data thinning
- (ii) curve smoothing
- (iii) slant removal

Many papers broach the problem of preprocessing techniques, and a number in some detail [3],[4].

Data thinning was only found to be necessary for the removal of redundant data produced when a user pauses with the pen tip resting on the paper, producing a string of proximate positional points. Otherwise, the tablet transmission speed was adjusted so as to capture sufficient data for a very fast writer.

Curve smoothing was found to be unnecessary for the tablet as it gave a sufficiently faithful character reproduction. However, the degree of smoothing is dependent on the tablet accuracy, which in a worst case could produce individual points well removed from the character path. In such instances, curve smoothing was performed by an iterative point averaging technique.

Slant removal was investigated and implemented. However, at a later stage it was found to be easier to accept slanted characters and cater for them in the databases.

2. THE X-Y ALGORITHM

The original technique of x-y travel analysis was particularly simple in its implementation. A paper by Tang, Tzeng and Hsu [5] extracts the x and y turning points in order to identify the numeral identities 0-9. This was used as the basis for the analysis of the relative x and y movement between the turning points. As a character is being formed x and y co-ordinate pairs identifying the curve path are monitored. Each time a transition from a positive to negative or a negative to positive travel is detected in either the x or y direction the distance travelled in that particular direction is noted. When the stroke is complete the positive and negative travels are normalised so that each increment of travel is represented as a percentage of the total travel in that direction. Very minor travel trends are filtered out. These are usually due to poor tablet accuracy or user indecision. Consider character 'a' as shown below:-

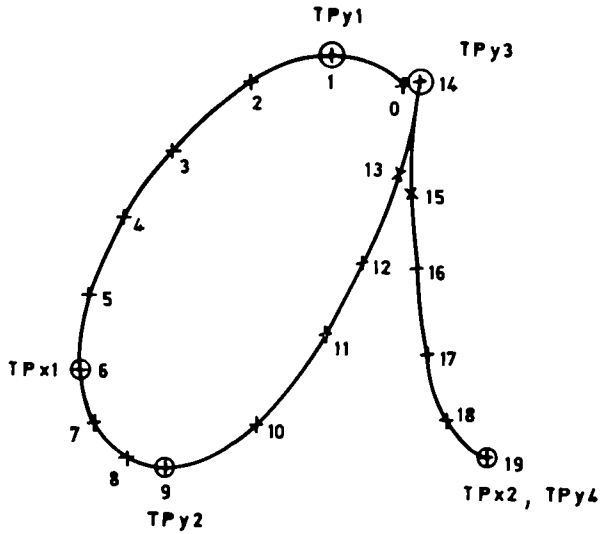


Figure 1 X-Y encoding of character 'a'

x - travel breakdown = (x_6-x_0) , $(x_{19}-x_6)$

y - travel breakdown = (y_1-y_0) , (y_9-y_1) , $(y_{14}-y_9)$, $(y_{19}-y_{14})$

$$\text{Total x-travel} = \sum_{n=1}^{19} |x_n - x_{n-1}| = X$$

$$\text{Total y-travel} = \sum_{n=1}^{19} |y_n - y_{n-1}| = Y$$

$$\text{Therefore, 'a'} = \left[\frac{(x_6-x_0)}{X} , \frac{(x_{19}-x_6)}{X} \right] , \left[\frac{(y_1-y_0)}{Y} , \frac{(y_9-y_1)}{Y} , \frac{(y_{14}-y_9)}{Y} , \frac{(y_{19}-y_{14})}{Y} \right]$$

Producing typical values:-

$$x = -0.45, +0.55$$

$$y = +0.05, -0.30, +0.30, -0.35$$

The lower case character set (a-z) was encoded using this technique in order to determine how unique each character was made. Initial analysis of a test set of 24 users showed that the technique was quite able to recognise the more complicated character strokes (eg. m,w,g,z,k) but performed poorly when attempting to recognise the simpler strokes (eg. -,/,c,l,j). In such

instances the algorithm was unable to distinguish the more subtle features of these simple strokes.

In order to extract more information from these simpler strokes, we decided to correlate the x and y travels in tandem. Thus, each time either the x or y travel is observed to reverse its direction, the incremental travel in both directions is noted. Therefore our encoded character 'a' would become:-

$$\text{'a'} = \left[\begin{array}{c} \frac{(x_1-x_0)}{X}, \frac{(x_6-x_1)}{X}, \frac{(x_9-x_6)}{X}, \frac{(x_{14}-x_9)}{X}, \frac{(x_{19}-x_{14})}{X} \\ \frac{(y_1-y_0)}{Y}, \frac{(y_6-y_1)}{Y}, \frac{(y_9-y_6)}{Y}, \frac{(y_{14}-y_9)}{Y}, \frac{(y_{19}-y_{14})}{Y} \end{array} \right]$$

One immediate benefit is that we can now reconstruct the character shape from the encoding. This is particularly useful for checking the XY database for bad entry encodings.

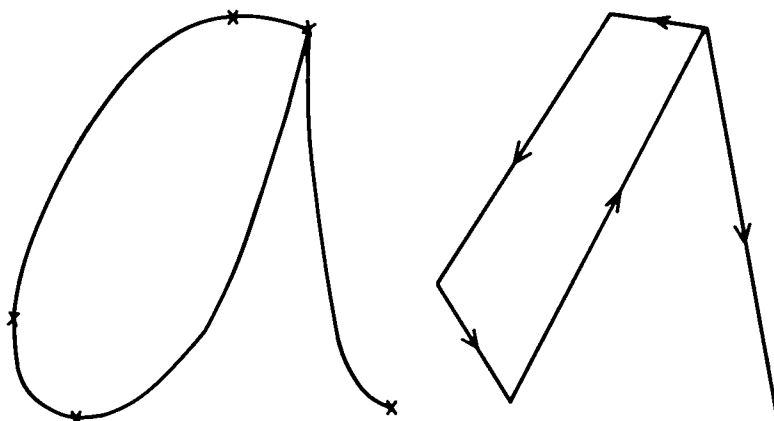


Figure 2 XY encoded character 'a'

This modification was found to increase the recognition rate, especially on the simpler strokes, although the recognition rate on the simple strokes was still unacceptably low. The algorithm could identify the character sub-group quite easily (eg. 'a', 'd', 'q' or 'h', 'n', 'u') but in order to improve the recognition rate another level of group-specific feature extraction would be needed. A subsequent search for another algorithm which would give more success with the simpler strokes was sought.

3. FREEMAN ALGORITHM

The idea of analysis of the travel path was quite appealing in its effectiveness and simplicity. The technique of curve path encoding by Freeman vectors appeared to be a natural progression from the XY algorithm. The idea is fully detailed in the paper by Herbert Freeman [6]. The advantage over the XY algorithm is that the curve path can be more closely followed. This means a better distinction between the simple stroke characters. The path of the curve is quantised into a number of vector directions. The greater the number of vectors, the more closely the curve can be tracked. We have started by selecting 8 vector directions, although one or two papers researched have selected up to 24 vector directions [7].

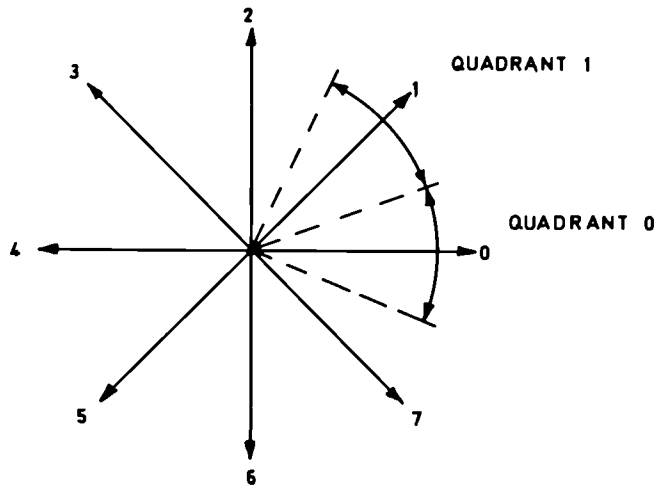


Figure 3 Freeman Encoding Vectors

While the angle of travel of the character curve remains in the bounds of a particular quadrant, determined from point to point, it is said to be travelling in the direction of the vector associated with that quadrant. Hence we can encode our character 'a' as follows:-

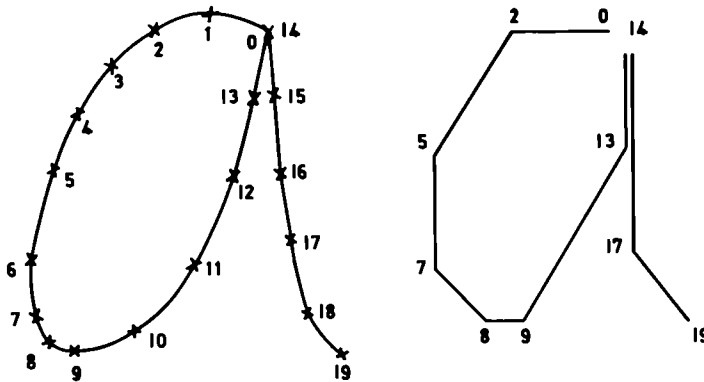


Figure 4 Freeman Encoding of character 'a'

The Freeman encoded string for the character 'a' is thus:-

$$'a' = '456701267'$$

As with the XY algorithm, the total travel is determined (in this case as a single component, T):-

$$\text{Total travel} = \sum_{n=1}^{19} [(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2]^{1/2} = T$$

We can then determine the contribution to the total travel for each of our vector directions:-

$$'a' = 4 \left[\frac{d_{0-2}}{T} \right], 5 \left[\frac{d_{2-5}}{T} \right], 6 \left[\frac{d_{5-7}}{T} \right], 7 \left[\frac{d_{7-8}}{T} \right], 0 \left[\frac{d_{8-9}}{T} \right], 1 \left[\frac{d_{9-13}}{T} \right], 2 \left[\frac{d_{13-14}}{T} \right], 6 \left[\frac{d_{14-17}}{T} \right], 7 \left[\frac{d_{17-19}}{T} \right]$$

Characters with similar shapes may produce strings of similar vectors (for example 'a', 'd', 'q'). Whenever a number of alternatives are produced it is possible to disambiguate the alternatives by a comparison of the relative vector travels of these alternatives to the unknown string.

From our initial sample data set of 24 users we produced a database of all the occurrences of Freeman vector strings derived from the character set a-z. The number of vectors per string was found to depend on:

- individual user writing speed
- character complexity

Any number of vectors from 1 to 20 could be produced. The larger the vector string:

- the more complex the subsequent decoding
- the larger the database allocation space required

It was decided to limit the number of allowable vectors per character curve to 5 by performing a number of vector reductions on the original vector string. Neighbouring vectors are compared in order to determine whether they could be candidates for possible combination, depending on relative size and angular variation. Our example character would reduce to five vectors thus:-

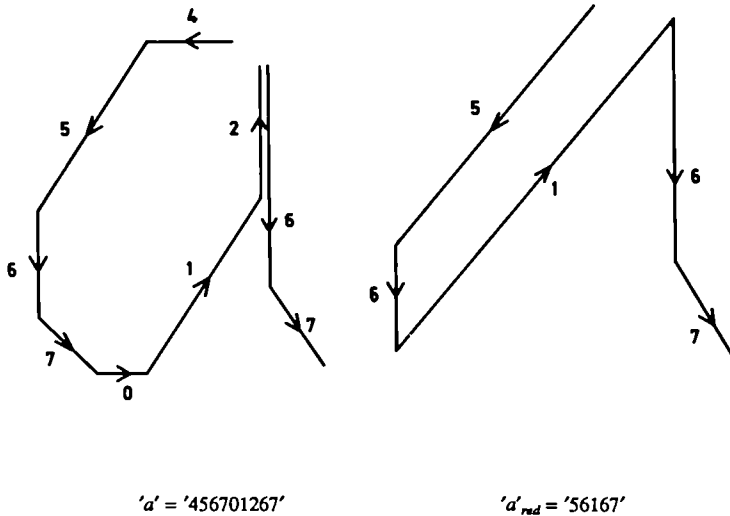


Figure 5 Freeman Vector Reduction

By limiting the maximum string length to 5 vectors we can limit the size of the Freeman database to a manageable number of unique vector encodings. Five vectors means that there are only around 22000 possible vector permutations, while allowing 8 vectors (which was done originally) gives around 6600000 possible vector permutations.

4. STROKE MATCHING

Because a user is allowed to write in a totally natural style, it is often found that in some instances characters are not formed completely in the time sequence. The characters 'i', 'j', 'f'

and 't' in particular are mostly finished off at the end of a word (and in some instances at the end of a sentence). In such cases we cannot assume a match against the last stroke. We must find the closest previously written stroke and analyse the relative shape and orientation of the two strokes in order to identify the composite stroke. Letters 'i','j','f' and 't' are by far the most common occurrence of two stroke characters, but it has been observed that virtually every other character in the lower case alphabet has, on some occasion, been formed using two or more strokes in our data set of 112 writers. Only the characters 'c','l','o','r' and 's' have so far only been created using a single stroke. Characters 'k','m','w' and 'z' have been formed with more than two strokes.

The output of the recognition phase is passed on to the stroke matching algorithm. A stroke being passed onto the matching algorithm is fitted onto the page in relation to the strokes which have been written prior to it. If the stroke is an extension of the current line, no matching is attempted. However, if it is sufficiently close to the previous stroke or if it is seen to be situated back along the present line or indeed on a previous line, then its position is compared with the positions of the previously written strokes along the appropriate line. If a sufficiently close partner is found a match is attempted. Analysis of two-stroke characters has shown that there is only a valid subset of allowable first strokes and, similarly, only a valid subset of allowable second strokes. If either the first or second strokes to be combined are not elements of the appropriate subset, then no match is attempted. Otherwise stroke matching is performed. This is done by using the identity of the first and second strokes as row and column identifiers in a character matching array. The corresponding array element indicates the result of the match. If a NULL indicator is the result, then it is not possible to combine the two strokes to make a sensible composite character.

CHARACTER MATCHING ARRAY											
	Second Stroke										
	[\]	-	.	/	c	l	z	o	e
[x	x	f	<	f	^	~			
\	x		*	t	<	!	^	!		*	
]	x	x		f	<	x	x	x		*	
/	k	!	*	t	<	t	^	~		*	
c		a	g	t	<		^			*	
l	^	t	*	t	<	@	^		^		^
r		^		f	<			^			
s		^		f	<			^			
v		^		t	<			^			
e				t	<	t		d			
b		k		b				-			
t		k									
o		a									
f				f							
z				z							
-						x					
.			j			k					
t		k									
First stroke											

AMBIGUOUS MATCHES:

* : y, b, p

^ : k, NULL if second stroke taller than first stroke

< : i, NULL if second stroke below first stroke

! : y, v, x

@ : y, t

~ : a, q, d

Figure 6 Character Matching Array

contained in the database. Using this technique, at most only 10% of the database need be searched for any particular unknown encoding.

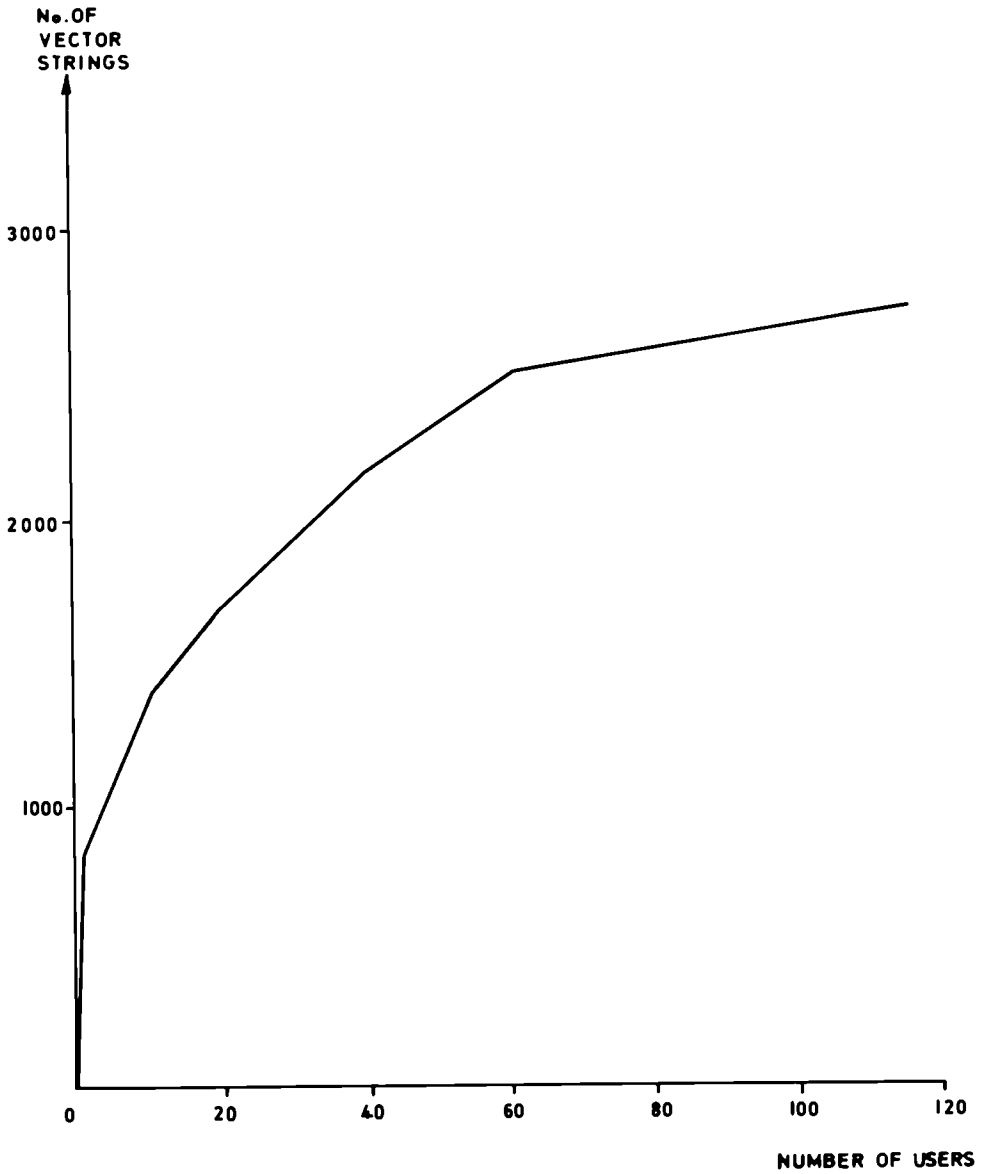


Figure 8 *Freeman Database Construction Profile*

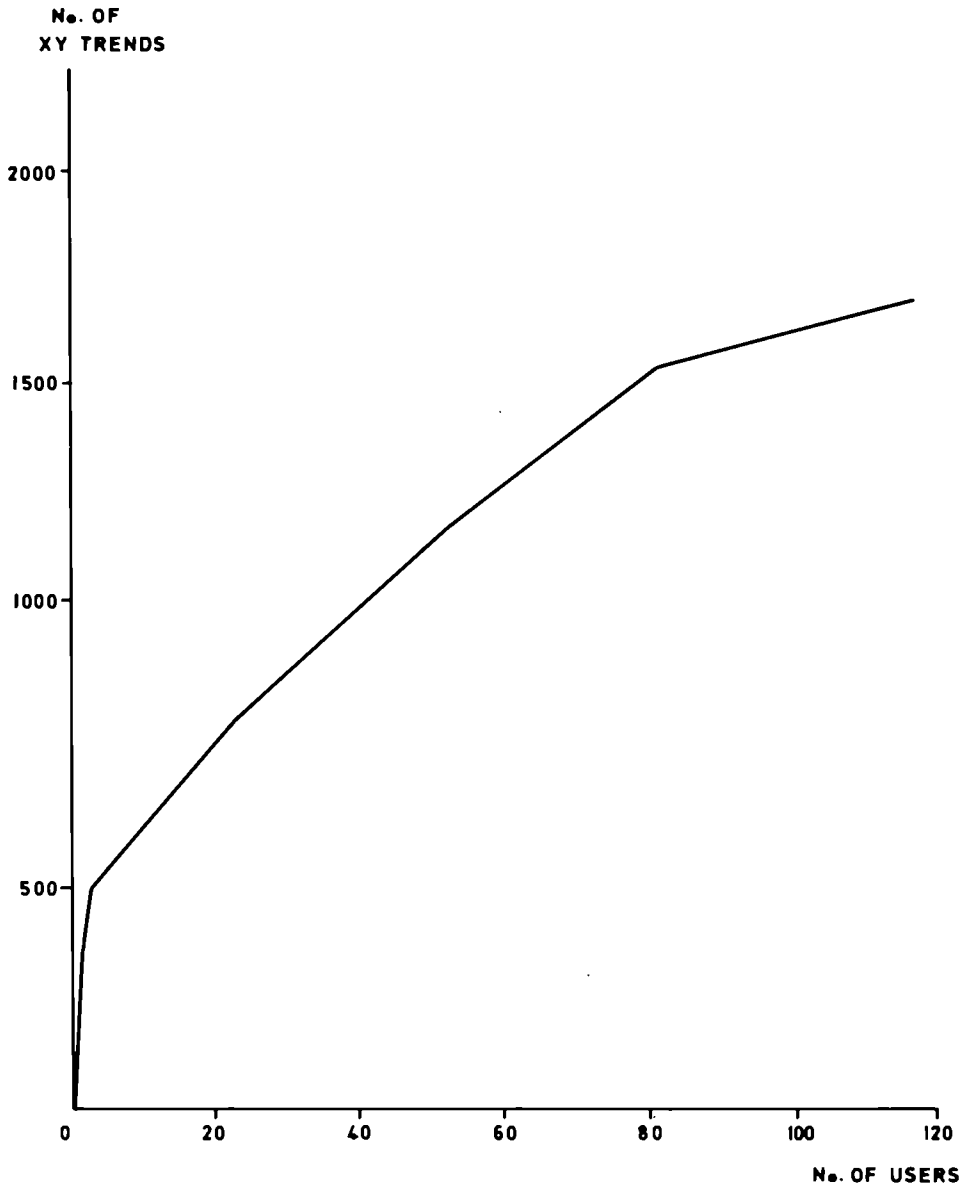


Figure 9 XY Database Construction Profile

6. RESULTS

At present the results of recognition performance are restricted to the 112 user database. The recognition rate on the isolated strokes was found to be:-

XY Algorithm - Recognition Rate = 77%

Freeman Algorithm - Recognition Rate = 93.5%

The XY algorithm was still particularly bad at differentiating the simple strokes in isolation. In particular, it confused the 'l', '/', '\', and '-' strokes. However, it could more easily recognise the more complex strokes, such as 'w', 'm', 'z', 'g'. Individual results shown below:-

Individual Results for XY Algorithm	
Character	Percent Recognition
l	35.35
/	26.92
\	40.24
-	23.71
w	95.87
m	94.50
z	96.94
g	93.69

The Freeman algorithm showed no such deviation in individual results:-

Individual Results for Freeman Algorithm	
Character	Percent Recognition
l	83.14
/	76.92
\	86.59
-	85.57
w	97.25
m	94.50
z	98.25
g	98.11

By correlating the results from the individual algorithms the overall recognition rate was found to be 95.20%, only slightly higher than the Freeman algorithm results. On our example characters, we get:-

Combined Results (Freeman and XY)	
Character	Percent Recognition
l	88.53
/	90.22
\	90.44
-	91.28
w	99.54
m	98.62
z	99.56
g	99.37

The next stage was the matching of the individually recognised strokes where appropriate. This introduced a further source of error. The major source of error being incomplete characters (especially undotted 'i's and 'j's). The other main error being the combination of the wrong strokes when a number of multi-stroke characters have been sequentially formed very close together. It is easy for the human brain to combine the correct pairings, since it uses dictionary search, context etc. After stroke matching:-

Recognition after matching - 90.00%

One process it was possible to perform at the end of the sentence was the tidying of the incomplete ('i' and 'j') and ambiguous ('/' and '\') strokes. It is possible to determine the proper character identity by analysis of the relative character sizes and orientations to the baseline. After this process, the final recognition rate was:-

Final recognition rate - 93.82%

The main sources of character mis-recognitions could be isolated into one of the following categories:-

- poor character input (due to a bad user writing style)
- poor character input (due to incomplete tablet data)
- incorrect database matching
- incorrect stroke matching
- incomplete character formation

This level of recognition, however, would not be acceptable to most users of a dynamic script input system. A higher recognition rate could be achieved by constraining the character positions and style, but this is contrary to the idea of a totally natural environment for user data entry. The alternative is to perform some higher level analysis or post-processing. As a result of the need to achieve a more acceptable recognition rate, Trent Polytechnic joined the project in January 1987 to investigate the application of intelligent knowledge based techniques to the input and decoding of the recognition results produced at this level. The Trent work will address the following aspects:-

- (i) letter adjacency rules
- (ii) knowledge bases for dictionary analysis
- (iii) word association rules
- (iv) sentence construction rules
- (v) contextual analysis rules
- (vi) concurrent analysis of ambiguities

7. CONCLUSION

The algorithms developed have produced good results on the limited user set obtained to date. However, a wider user base is being sampled in order that a better determination of the user independence of the technique can be made. This will enable us to validate the completeness of the x-y and Freeman databases and also the stroke matching array. Although the Freeman technique gives far better overall results for unconnected characters, the x-y algorithm will also be retained for the analysis of cursive script. Much of the x-y technique misrecognition on 'good' characters was found to be due to the characters having 'toes' and/or 'tails' which confuse the x-y algorithm if they are too large. It is felt that the x-y technique will be more successful on the recognition of the character shape with all the ligatures removed.

Presently, characters are displayed as they are recognised on the SUN monitor. We need to determine the impact of the post-processing of the text which is being undertaken by Trent Polytechnic.

- how will the post-processing techniques affect the overall speed of the system?
- at what stage should any corrections from the post-processor be indicated to the user (character level, word level, sentence level)?

Work is also being undertaken to allow the user to perform some natural editing functions as the user might perform when writing on a sheet of paper. This involves the recognition of some strokes as editing symbols as opposed to alphanumeric characters. The initial functionality of the editor has been kept very simple. It allows the user to:-

- change a character by writing a new character directly over the old one.
- delete a character or number of characters by drawing two horizontal lines through the characters to be deleted.
- insert a character or number of characters. The insert position is marked by marking an insert symbol between the appropriate characters or words. The string of characters to be inserted can be written anywhere on the paper. End of insert is detected as a return to the continuation of the current sentence.

Other features which would be useful for natural editing include:-

- text block moving
- paragraph marking
- text block deletion

It is not feasible, with the demonstrator in its present form, to implement an editor with more functionality, as it would become particularly confusing for a writer when working between the

tablet and screen. A transparent tablet situated over a flat screen would provide a better physical interface and technology is rapidly progressing towards making this a feasible solution in the next year or so.

Work on cursive script recognition is starting and will continue for the duration of the project (September 1989). In particular it will concentrate on the feasibility of user independent script recognition systems.

REFERENCES

- [1] On-Line Cursive Script Recognition
M. Berthod and S. Ahyon
Proceedings of 5th Int. Conf on Pattern Recognition
Vol 2 pp 723-5 IEEE 1980
- [2] Machine Recognition of Roman Cursive Script
S. Badie and K. Shimura
Proceedings of 6th Int. Conf on Pattern Recognition
Vol 1 pp 28-30 IEEE 1982
- [3] A Normalising Transform for Cursive Script Recognition
D.J. Burr
Bell Labs, IEEE Transactions pp1027-1030 1982
- [4] Preprocessing Techniques for Cursive Script Recognition
M.K. Brown and S. Ganapathy
Computer Graphics and Image Processing pp447-458 1983
- [5] A Microcomputer System to Recognise Handprinted Numerals Using a Syntactic - Statistic Approach
Tang, Tzeng and Hsu
7th International Conf. On Pattern Recognition
Vol II pp 1061-4 1984
- [6] On the Encoding of Arbitrary Geometric Configurations
H. Freeman
IRE Transactions on Electronic Computers pp260-268 June 1961
- [7] On-Line Recognition of Hand-Written Characters Utilizing Positional and Stroke Vector Sequences
Katsuo Ikeda et al
Pattern Recognition Conference 1980
Vol 13 No 3 pp 191-206

Project No. 291/860

LINGUISTIC ANALYSIS OF THE EUROPEAN LANGUAGES

V. VITTORELLI

Ing. C. Olivetti & C., S.p.A., Voice Processing Lab.
Cso Svizzera 185, 10149 Torino, Italy.

The paper describes the purpose of the project and the expected results.

The purpose is to make a set of computer usable models, based on a common methodology, of seven european languages: Dutch, English, French, German, Greek, Italian, Spanish.

This set of models is to include databases, statistics, algorithms useful for speech processing systems and for other tasks related to the natural language processing.

The expected results are: a common phonetic alphabet, and, for each language, a dictionary providing the graphemic and the phonemic representation of the words along with their grammatical categories and frequencies; the frequencies of letters and phonemes and the frequencies of letter and phoneme clusters; a list of homographs (words with the same alphabetic form and different pronunciation) and of homophones (words with the same pronunciation and different alphabetic forms); the distribution of word length in alphabetic and phonemic form in the dictionaries and in the textual data bases; a set of rules for grapheme to phoneme conversion and for phoneme to grapheme conversion written using a common formalism; a set of grammatical category tags; sequence matrices providing the observed frequency of any possible pair of properly defined grammatical categories; a common algorithm that, using the language specific rules, accomplishes the phoneme to grapheme conversion; a common procedure for solving the ambiguities due respectively to homography and to homophony in grapheme to phoneme conversion and in phoneme to grapheme conversion.

1. FOREWORD

The increasing sophistication of the speech processing systems requires and relies more and more on linguistic knowledge.

Based on this assumption that will be justified in the following, we have proposed to the ESPRIT management the project 291 (later 860) with the goal of collecting databases, statistics, algorithms, for modeling seven european languages in a number of aspects definitely or potentially relevant for designing and for testing text-to-speech and speech-to-text conversion systems.

The seven languages that are currently analysed in this project are: Dutch, English, French, German, Greek, Italian, Spanish (in alphabetic order).

2. THE CONSORTIUM

Given the goals of the project, it was obviously wise to have a partner for each language, located in the corresponding country, having as far as possible experience in this field and motivated to address this analysis.

Since, at the time when the project started, Spain was not a member of the European Community, the analysis of Spanish was committed to an Italian institution but, as soon as possible, a Spanish partner took over this part of the job according to the original vision of the natural partitioning of this project.

Simultaneously, a Greek partner was invited to undertake the analysis of the Greek language not included in the original project.

The enlargement of the project from six to eight partners was also an opportunity for including in our plans some additional analysis and this extension of the Consortium and of the goals marks the difference from the original project 291 and the current 860.

The Consortium composition is the following:

Ing. C. Olivetti & C., S.p.A. - Prime Contractor - ITALY

Acorn Computers - GREAT BRITAIN

C.S.A.T.A. - ITALY

Katholieke Universiteit Nijmegen - THE NETHERLANDS

L.I.M.S.I. - FRANCE

Ruhr Universitaet Bochum - GERMANY

Universidad Nacional de Educaci3n a Distancia - SPAIN

University of Patras - GREECE

3. THE ROLE OF LINGUISTIC KNOWLEDGE IN SPEECH PROCESSING SYSTEMS

The relevance of linguistics in the design stage and in the testing stage of the speech processing system has been increasing over the years and is expected to increase further in the future particularly for the systems intended for conversion of a free text into speech and for the systems intended for the reciprocal transformation: conversion in written text of a sentence uttered by a human being.

Linguistics has no relevance for other voice processing applications like pure voice mail systems, that are essentially based on signal processing algorithms.

3.1. Text-to-speech.

For those who are not familiar with the speech technology, we summarize here the essential steps of the procedure necessary for text to speech conversion.

- . First, the input text has to be converted in a pure sequence of words: acronyms, abbreviations, numbers, any information other than plain words, has to be converted in a sequence of words. This is a kind of linguistic knowledge that is not part of the analysis done within the ESPRIT project.

- . Each word, has then to be converted from the usual graphemic form into a phonemic form, inclusive of a stress mark.

This step of the procedure is obviously language specific and it is worth to note that in many languages there are cases of ambiguity due to homography. Examples are the English word "read" that is pronounced in two different ways in the sentence "I will read the document that you have read yesterday"; the French word "couvent" (les poules du couvent couvent); the Italian word "ancora" (la catena dell'ancora si è rotta ancora) etc.

As it will be discussed in the following, project 860 has to provide the algorithms for the grapheme to phoneme conversion (inclusive of stress location), lists of ambiguities due to homography, and means for ambiguity solving.

- . The next step is to define the sentence prosody: pauses, rythm, pitch profile, word deaccentuation.

For a good emulation of natural speech, all of these prosodic features should be derived from a full syntactic and semantic analysis of the sentence.

Project 860 does not provide the means for that, but it provides the means for identifying the phrases that appear in a sentence.

This requires to give the correct tag to the functional words that can belong to different grammatical classes like for instance the word "la" that in French, Italian, Spanish can be either an article or a pronoun.

Project 860 provides lists of (frequent) words with the respective grammatical tags and means for solving the grammatical ambiguities based on the local context, in form of a Markovian model and disambiguation rules.

- . Next step is to replace the phonemic string with a sequence of parameters suitable for driving a sound synthetizer.

This is always done using a catalogue of basic sound units (in parametric form).

These units can be the basic phonemes, or sequences of two phonemes (diphones), or any cluster of phonemes, or words. Each symbol (or symbol sequence) of the phonemic string will be replaced by the suitable sequence of sound units to be smoothly connected with rule generated transients, and durations.

The problem of allophonic variations can be solved using diphones or phoneme clusters as basic sound units.

Project 860 provides the frequence of use of phonemes and phoneme clusters so that the system designer has the basic information for deciding wich sound units should be included in the basic sound catalogue.

- . Finally, the parameter string is fed to the sound synthetizer.

3.2. Text dictation.

The traditional approach to speech recognition that requires each speaker to train the system providing one or more tokens of the words that he (or she) intends to use, is not suitable for the dictionaries of several thousands words necessary for text dictation.

In our opinion, a system for text dictation should have among the others an important feature: the enrollement of a new speaker should be done with a standard procedure, independent from the dictionary to be used and (of course)

should be as simple as possible; conversely the dictionary definition or modification should be done by keyboard, providing the graphemic form, and this should not require a new adaptation to the speakers voices.

As a consequence, the system should have either the capability of converting the graphemic form into some kind of phonemic representation as it is necessary for comparing the input sound with the word models, or the system should be able to convert the input sound into a phonemic string and in turn the phonemic string in a set of possible grapheme strings for a search in the dictionary.

The first solution is the most usual, the second one would present the advantage to use the dictionary in one form (graphemic) instead of two (graphemic and phonemic).

Within project 860 we provide the algorithms for grapheme to phoneme conversion (necessary for the first solution) and we investigate the feasibility and the convenience of a phoneme to grapheme conversion.

Any recognition system intended for dictation and using as a consequence a large dictionary, has to propose a lattice of word candidates. This is due in principle to the existence of homophones, but in practice the limited performances of the existing acoustic analysers, the consequences of possible mispronunciation and disturbing noise etc., suggest to propose a lattice more than a string of single words.

Of course, the lattice will be more complicated if the system is intended for continuous speech, simpler if it is intended for isolated words.

Project 860 provides lists of homophones, and the basic knowledge for selecting a likely word sequence from the lattice after a contextual analysis.

This is the purpose of the sequence matrices on grammatical categories, and of the binary and ternary rules, that will be discussed later.

An important aspect of this approach is that it relies on the grammatical categories only. As a consequence, the sequence rules can be applied to any new word provided that the grammatical tags are available.

4. ANALYSIS PROCEDURE OUTLINE

4.1. Data processing standards.

As a first step, the Consortium decided to address the definition of a set of common data processing standards.

Unfortunately, the partners had in house different kinds of equipment and different programming skills. Despite of this an agreement was found to use fully compatible mini computers and operating systems and high level programming languages (C, Pascal and for some pre-existing software modules or packages, Fortran).

4.2. Common Phonetic Alphabet.

The Consortium has defined a common "phonetic alphabet" in computer readable form.

It consists of a synoptic table providing the correspondence of the basic sounds (essentially at phonemic level) existing in the seven languages, to the symbols of the International Phonetic Association and to a set of computer readable symbols.

In addition, the Consortium has set a standard for representing the location of stresses (at different levels), and of the syllable boundaries.

4.3. Text corpora.

In order to provide a set of consistent results, the Consortium has selected as a basis for the analysis a set of corpora existing in six different languages. The source of the documents (recorded on magnetic tapes) is the EEC itself. Unfortunately the corpus did not exist in Spanish, and as a consequence the raw material is consistent for the other six languages while for Spanish it is a newspaper.

The first corpora consisted of 100.000 words each; subsequently the Consortium has agreed to analyse additional 300.000 words for each language (again, the source is the EEC) and for this second batch it will be possible to analyse seven omogeneous corpora.

Some of the results produced by the analysis (for instance the sequence matrices on grammatical categories) are expected to be to some extent dependent on the peculiarity of the raw material.

In order to evaluate the relevance of this problem, this material will be complemented by testing the sequence matrices on a totally different style: newspapers.

4.4. Texts labelling.

In order to provide the basis for the desired analysis, it is necessary to label the text corpora.

Given the context, each word is classified using the proper grammatical category.

For instance, in the sentence "it is necessary to label...", the word "label" has to be classified as a verb rather than as a noun.

This process has been done using a common, tree structured scheme that is unified for the broad classes and language specific for the subclasses (the tree leaves).

4.5. Dictionaries.

The dictionaries are representative of the text corpora. Each word (each different graphemic form) found in the corpus appears in the corresponding dictionary along with the grammatical tags (all the possible grammatical tags), the frequency of each different reading measured in the corpus, and the corresponding phonemic transcriptions inclusive of the stress mark.

4.6. Grapheme-to-phoneme conversion.

The dictionaries, are used as a reference for testing the grapheme to phoneme conversion.

The grapheme to phoneme conversion is accomplished by means of a set of software routines (one for each language) having a common software interface, intended for converting any plain word from the standard orthographic representation into our standard phonemic representation.

Another set of routines performs the conversion of a number expressed as a sequence of digits into the corresponding sequence of words to be fed to the

grapheme to phoneme conversion. For instance in German the number 142 will be transformed in the sequence "hundert-zwei-und-vierzig".

A full discussion of the different problems found in the seven languages in performing the grapheme to phoneme conversion would go beyond the purpose of this report.

Anyhow we can say that despite the commonality of many words and roots the problems are quite different. In this respect, English seems to be the most intriguing of the seven languages.

English is not more rich of homographs than the other languages, but even for the words that have a single pronunciation, a reliable conversion system has to be based on a morphological decomposition and on a long list of morphemes with their phonemic transcription.

The case is different for the other languages where a compact set of rules and a comparatively short list of exception words, is sufficient for achieving a satisfactory performance.

4.7. Grapheme-to-phoneme conversion performance evaluation.

The Consortium has defined a common, unified procedure for evaluating the performance in grapheme to phoneme conversion. The results are not yet available.

4.8. Phoneme-to-grapheme conversion.

The project includes an investigation on the feasibility and on the practical usefulness of a phoneme to grapheme conversion at word level.

While it was known in advance that this process could only produce a set of possible graphemic strings to be compared against a dictionary of existing words in graphemic form, it was not known how large this set and how long the rules list would be in the different languages.

Again, a complete discussion of the problems found in the different languages would go beyond the purpose of this report but it is worth to note that French is probably the language more difficult to treat with such a procedure, as the number of possible transcriptions is on average definitely higher than in other languages.

The phoneme-to-grapheme conversion by rules as opposed to a search in a list providing the phonemic forms, seems to be a viable procedure for Dutch, German, Greek, Italian and Spanish.

4.9. Statistics on the dictionaries.

The dictionaries are a main information source for statistic analysis providing the following results.

- . Frequency of graphemes, grapheme clusters and words in graphemic form.
- . Frequency of phonemes, phoneme clusters and words in phonemic form.
- . Homographs and homophones lists.
- . Word length distribution in graphemic and phonemic form.

Since the dictionaries provide also the use frequency of words in the corpora, all the statistics can be computed counting each word as a unit or weighting it with its frequency.

- . Text coverage. After sorting the dictionary by decreasing use frequency, it is possible to compute the percentual coverage of the corpora as a function of the number of forms considered.

4.10. Markov model of the languages.

The labelled corpora are the basis for computing the transition matrices on grammatical categories.

Essentially, they provide the observed frequency of any pair of grammatical tags. As an example, the matrices will show (in a quantitative way) that an article is frequently followed by a noun or an adjective and seldom (or never) by another article or an adverb.

The effectiveness of the system depends on the number and on the definition of the grammatical categories.

In principle, many, small and well defined categories should provide a more accurate sequence model than few, large classes, but in practice, the problem of making a statistically significant evaluation of the transition probably might offset the benefit.

The Consortium has decided to use rather fine categories but there is the option of defining the so called "cover symbols" for defining large classes as the conjunction of a set of smaller classes.

The actual definition of the categories within a unified frame is language dependent and will be established by the linguists of the various group probably after some iterations.

Of course, the larger is the corpora, the better will be the matrices estimation.

In order to speed up the process of labelling the corpora, the Consortium has adopted a semi-automatic labelling procedure that relies on a preliminary estimation of the matrices.

The additional labelled texts obtained in such a way are then used for a better estimation of the sequence matrices and so on.

4.11. Binary and ternary disambiguation rules.

In addition to the Markov models as described above, the Consortium intends to identify and use binary and ternary disambiguation rules that can be extracted from the corpora and that reflect additional constraints on the sequences of grammatical categories in natural language.

5. SOFTWARE

The analyses mentioned above require a very significant amount of software.

In order to limit or avoid possible duplications, the software units have been classified as language independent or language specific. The language independent software units have been analysed and implemented by small teams of one or two partners and are or will be used by all the Consortium members.

6. CONCLUSION

The project started in Nov. '84 (as 291) and will end in Dec. 1988 (as 860). It is too early for stating to which extent it will be successful but so far we believe that all the targets will be reached.

The size and the choice of the corpora used in building the Markovian model can be seen as a possible limitation of the validity of the results, but the existence of the tools for automatic labelling and sequence matrices generation and handling, means that additional corpora can be analysed in a comparatively easy way.

The prime proposer is glad to have an opportunity for acknowledging the high professionalism of the members of the Consortium and thanking all of them for the cooperation.

BIBLIOGRAPHY.

- . Proceedings of IEEE, Nov. 1985
Special issue on man machine communication (a very rich bibliography can be found in this issue).
- . D.W. Shipman and V. Zue "Properties of large lexicons: Implications for advanced isolated word recognition systems", in Proc. ICASSP 1982
- . R. Carlson, K. Elenius, B. Ganstrom, and S. Hunnicutt, "Phonetic and orthographic properties of the basic vocabulary of five European languages", Speech Transmission Lab Quart. Progress Rep., STL.- QPSR 1-2, 1985.
- . F. Jelinek, R.L. Mercer, L.R. Bahl, and J.K. Baker, "Perplexity-A measure of difficulty of speech recognition tasks", presented at the 9th Meet. Acoustical Society of America, Miami Beach, FL, Dec 15, 1977.
- . C.E. Shannon, "Prediction and entropy of printed English", Bell Syst. Tech. J., vol. 30
- . The linguistic Processor in a Multi-Lingual Text-to-Speech and Speech-to-Text system, L. Boves and M. Refice, Nijmegen University, The Netherlands and CSATA, Italy.**
- . Contextual Syntactic Analysis for Text-to-Speech Conversion, S. Quazza and E. Vivalda, Ing. C. Olivetti, Italy.**
- . "Phoneme to Grapheme Conversion by Rules"
W. Senders; R. Willemse and W. Bloemberg, Nijmegen University, The Netherlands.
- . "Phoneme to Grapheme Conversion System for Unrestricted German Vocabulary"
U. Jekosch, Ruhr Universitat Bochum, FRG.**
- . "Improving Text to Speech Conversion in Spanish: Linguistic Analysis and Prosody"
J.M. Pardo, M. Martinez, A. Quilis and E. Munoz, ETSI Telecomunicaciòn and Univ. Nacional de Educaciòn a Distancia, Spain**
- . "Syrib" Text to Speech System for Unrestricted German Text,
M. Kugler-Kruse and R. Posmyk, Ruhr Universitat Bochum, FRG. **

** To be presented at the "European Conference on Speech Technology", Edinburgh, Sept. 1987.

Standardized Interchange Formats for Documents

Günther Krönert

*Siemens AG, Data Systems Division,
Application Programs, Munich, Germany*

Interchange of documents is an important part of office communication. Documents which are interchanged in electronic form should not only be printable at the receiver's site, but it should also be possible to edit, reformat and process received documents in the same way as at the sender's site. In order to allow such an interchange between open systems, a standardized Office Document Architecture (ODA) and Office Document Interchange Format (ODIF) is being developed by the international standardization committees, like ECMA, ISO, CCITT. The significant properties of ODA are: A document has a logical and a layout structure, it is an instance of a definable document class and may have different content types, such as text, raster graphics, etc. This paper also describes how Office Document Interchange Formats on different levels can be derived from ODA and reports about first prototyping of an ODA editor within the Project 121. An overview on the further development of ODA is given at the end of the paper.

1. THE NEED FOR STANDARDS

Office automation consists in the integration and support of various office functions by electronic systems. These office functions include: Word-processing, electronic filing, electronic mail, voice mail, administrative support and decision support. Communication is necessary to integrate office functions. Communication, i.e. the exchange of information, originally only took place between people. It is possible to differentiate between direct communication such as discussions or telephone conversations which allow a spatial distance between the partners but not a temporal one, and indirect communication which allows both spatial and temporal distances. When observed from an abstract standpoint, the bearer of indirect communication is the document: Partner A draws up a document in accordance with conventions (e.g. characters, language etc.) known to both partners, which expresses the information to be conveyed. Partner B can interpret the document at a later date in accordance with these conventions, thereby obtaining the conveyed information. If these conventions or parts thereof are also "understandable" to machines, it is possible to a certain degree for man to communicate with systems or systems to communicate with other systems by the interchange of documents /1/.

The document is therefore the main information medium in the office and a key aid in the integration of office functions /2/. However, the progress made by microelectronics has changed the form of the presentation from a "paper document" to an "electronic document". This means that, in principle, documents can be transmitted "electronically" and the receiver can print them out or display them on his screen and process them further, or they can be interpreted automatically by the receiver and incorporated into office processes. However, in order to ensure that this is also possible outside a closed system, standardized interchange formats are required for such documents. For a number of years a systematic approach has been developed for such interchange formats in the international standardization

committees. First of all, a model was developed which describes how documents are structured. The standards refer to this model as ODA (*Office Document Architecture*). A family of *Office Document Interchange Formats* (ODIF), which define the coding for documents to be interchanged electronically, can be derived from this model.

ODA/ODIF is discussed and developed for several years in a number of international standardization bodies. First it was passed by the ECMA as ECMA Standard 101 (September '85). This standard was amended and extended to the ISO/DIS 8613 (Draft International Standard) in April 86 /3/, which is expected to become an International Standard in November 87.

2. THE DOCUMENT ARCHITECTURE MODEL ODA

The document architecture model is intended to be general and forward-looking. It should make it possible to construct simple text editors as well as future, interactively formatting, editors for documents with mixed information and mixed media /4/.

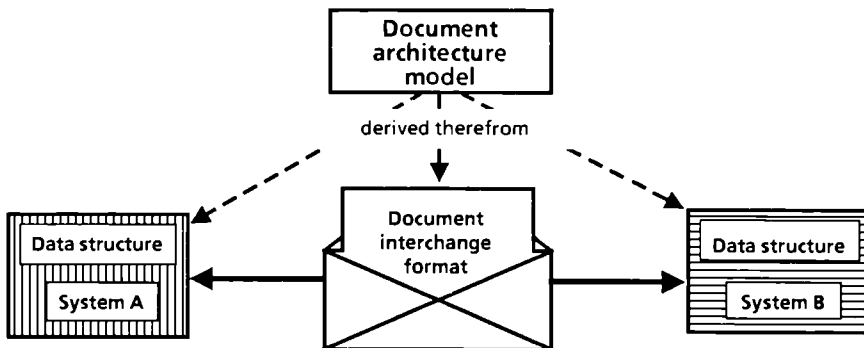


Figure 1: Significance of Document Architecture Model

Editing and formatting a document are processes which are much more complicated than performing a simple reproduction /5,6/. For this reason, the development of the document architecture model laid great store on ease of processing. Document processing systems have their own internal data structures for documents. These are determined by the operations which are executed on them. In order to transmit documents, these internal data structures must be converted to interchange formats (Figure 1). Ideally, these conversions should be possible without any loss of information. This is ensured if both the data structures and the interchange formats are derived from the same document architecture model.

2.1 Overview

A document has two structures, namely a logical structure, which subdivides the document into chapters, sections, sentences, images, etc., and a layout structure, which subdivides the document into pages and rectangular areas on the pages.

The document architecture model ODA /7, 8/ is based on the declarative approach, i.e. the structures are not expressed implicitly by control characters in the document content but are given explicitly by a hierarchy of objects /9/. In addition to the hierarchical relations between objects, non-hierarchical relations exist, e.g. references to footnotes. In addition, a large number of other characteristics such as size, dimension, alignment, etc., can be expressed in attributes.

It is vitally important for a clear interface to exist between the document structures and the document content (Figure 2). Only the basic objects can have a content, e.g.

character text, raster graphics, vector graphics, etc. This content is then structured in accordance with the appropriate content architecture. The standard scope can be easily expanded via this interface by adding further content architectures. The document structures are completely independent of the content architectures.

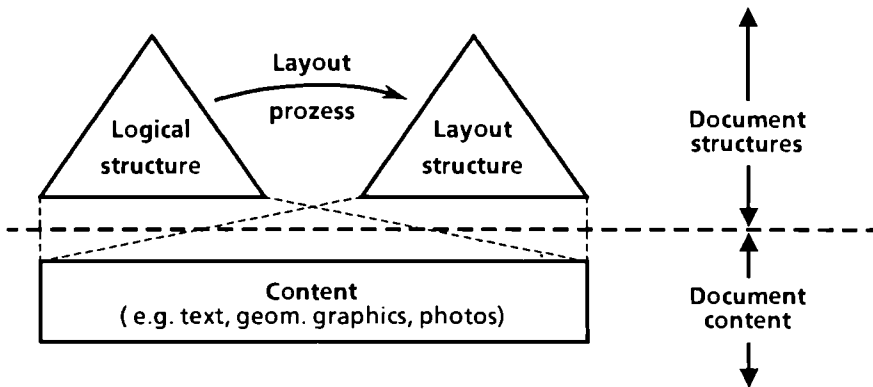


Figure 2: Document Structures and Document Content

Each logical object or layout object is an instance of an object class which has to be defined. Since the entire document, too, can be understood as an object, it is also an instance of an object class (also called document class). The definition of a document class consists of definitions of object classes and, possibly, of prescribed generic content for objects of certain object classes, such as logos and standard paragraphs. This class definition can be compared with a type of grammar or set of rules which can be used to construct individual documents of this class. Examples of document classes include report, business letter of company XY, business letter of company Z, order form, travel expenses form, etc.

2.2 Structures, Attributes and Content

In the model concept, each of the two structures is expressed by a hierarchy of objects. Each such object is defined by a certain object class and a certain object type. **Object types** are defined in the standard. This indicates which attributes can be used on these and which role they play in the document architecture. In the document class definition, **object classes** are defined on the basis of object types, i.e. object types can be regarded as superclasses of object classes. The standard provides means in order to be able to define such object classes in document classes.

ODA provides the following object types for constructing the logical structure:

- *Document Logical Root*. An object of this type is the first level in the logical structure, i.e. it can be regarded as the root of the logic structural tree.
- *Basic Logical Object*. These logical objects are on the bottom level of the hierarchy, i.e. they represent the leaves of the structural tree. They contain content portions of the document such as text or images.
- *Composite Logical Objects*. Objects of this type can be used to construct any number of hierarchical levels between the *Document Logical Root* and the *Basic Logical Objects*. However, they have no actual content.

ODA provides the following object types for the layout structure:

- *Document Layout Root*. An object of this type is the uppermost level in the layout structure, i.e. it can be regarded as the root of the layout structural tree.
- *Page Set*. A group of pages can be combined in an object of type *Page Set*.

- *Page*. An object of type *Page* is a two-dimensional area on which the content of the document is positioned and displayed.
- *Frame*. An object of type *Frame* describes a rectangular area on a page in which the contents can be formatted during layout design. This produces blocks in frames. In general, blocks are not allowed to occur outside *frames*.
- *Block*. The content of a block can only be of one type, e.g. either only text or only graphics. Blocks are the only layout objects which contain content direct.

These two structures break down the content of the document into content portions. Several *Basic Logical Objects* can be allocated to *Basic Layout Objects*, e.g. a chapter heading consists of a chapter number and the heading text. In the layout, both appear on the same line and, consequently, in the same *Block*. Several *Basic Layout Objects* can also be allocated to a *Basic Logical Object* if, for example, a section is divided into two blocks by a page break.

The content can be different in type, e.g. text, geometric graphics, photos, etc. For each type of content there is a corresponding *Content Architecture* which defines the structure, content-internal structures, control characters and coding, etc. These *Content Architectures* are not part of the document structures, but the document architecture provides a uniform interface between the structures and the content. This means that further content architectures are permissible without changing the document architecture. So far, a *Character Content Architecture*, a *Geometric Graphics Content Architecture* (based on CGM) and a *Raster Graphics Content Architecture* have been defined in ODA.

Attributes can be allocated to the objects of both structures to describe the characteristics of these objects. Each attribute is defined by a specific attribute type which defines the semantics of the attribute, i.e. the dimension, and has a value which is then interpreted accordingly, e.g. as the measures of the dimension. In total, currently approx. 40 different attribute types have been defined in the standard.

2.3 Document Class Concept

In order to be able to process a document as conveniently as possible, it is necessary to interchange information on the description of the document as well as information on the processing of the document. In the architecture model, the components of which are shown in Figure 3, this means that a document (*Specific Structures and Content Portions*) is viewed as an instance of a document class described in the *Document Class Description*. This consists of the three components *Generic Logical Structure*, *Generic Layout Structure* and *Document Style*.

The *Generic Logical Structure* describes all logical structures which are correct for documents of this class. It consists of a set of *Logical Object Class Descriptions* so that each object of the logical structure is understood as an instance of such a class.

A *Logical Object Class Description* consists primarily of a *Construction Rule* which specifies which subordinate objects an object can be constructed from. It can be compared with the production of a context-free grammar. The *Class Description* contains expressions which define how the attribute values of a generated instance of this class are to be calculated. When generating an instance, content can also be generated automatically with the aid of *Generic Content Portions*, e.g. the predefined content of a standard paragraph or a blank form.

The same applies, *mutatis mutandis*, to the *Generic Layout Structure*.

The *Style* defines how content and permissible logical structure are to be mapped onto one of the permissible layout structures of the document class. A *Layout Style* is a group of attributes controlling the automatic generation process of the layout structure. A *Presentation Style* is a group of attributes which supply the attribute values for describing layout objects. An example of an attribute from the first group is *Offset* which defines the minimum distances between a generated block and the

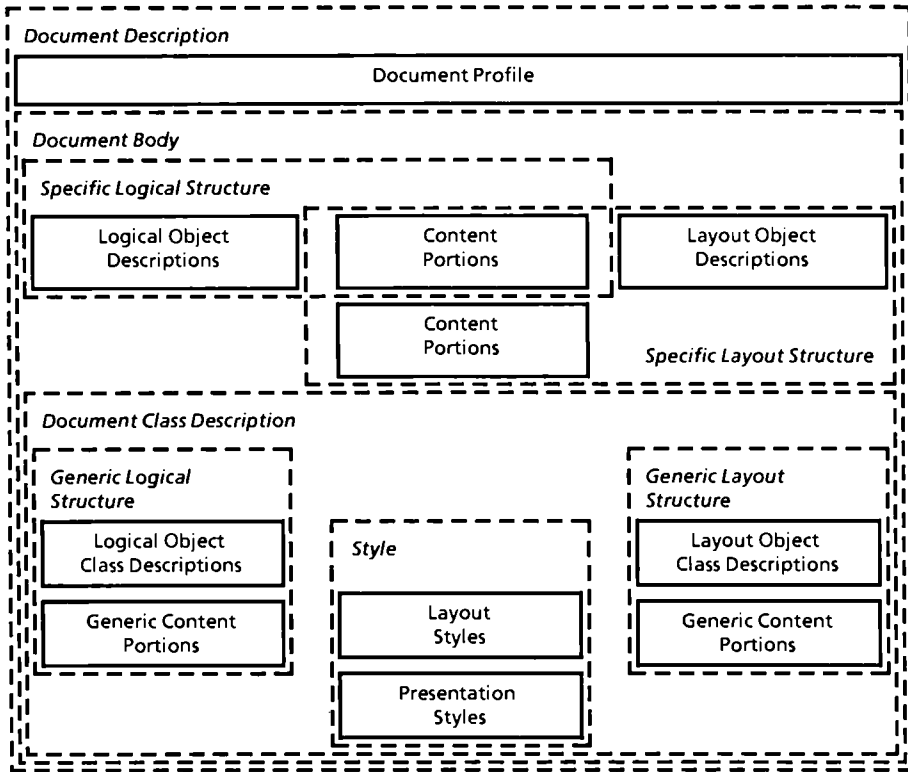


Figure 3: Components of Document Description

surrounding frame. Attributes of the *Presentation Style* include type face, line spacing, etc.; these attributes only control formatting of the content within a block.

2.4 Document Processing Model

The document processing model shows which components of the architecture control or influence which sequences. As Figure 4 shows, this model consists of 3 processes: the editing process, the layout process and the imaging process. However, the sequence of the processes in the figure does not mean that they must adhere strictly to this order in an implementation (this would correspond to batch processing). In an interactively formatting editor, they run cyclically.

The editing process calls for a differentiation between processing of the logical structure and processing of the content. Only such logical structures can be generated as can be produced from the *Generic Logical Structure*.

The layout process also consists of two cooperating processes, namely generation of the layout structure and formatting of the content into blocks. These two processes cooperate like co-routines. The basic principle of the layout process is such that the logic structure tree is run through in preorder and an attempt is made during this process to generate a layout structure tree (also in preorder) which is permissible in accordance with the *Generic Layout Structure*. This generation process is controlled by the *Layout Styles* which specify, among other things, which layout objects are to

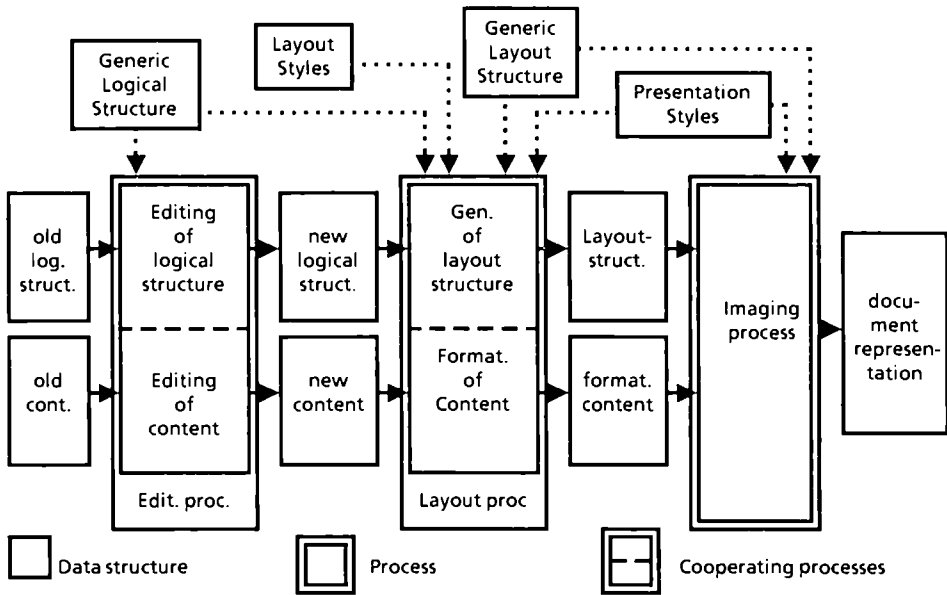


Figure 4: ODA Document Processing Model

be generated for which logical objects. The formatting of the content into blocks, too, can make demands on the generation process.

The imaging process can also access *Presentation Styles* and *Generic Layout Structure*, and in particular *Generic Content Portions*.

3. DOCUMENT INTERCHANGE FORMATS

The ISO 8613 standard defines the ODIF interchange format, i.e. how a bit stream for document transmission is to be generated which expresses the information of a document description conforming to ODA. This is the most powerful ODIF interchange format. In many cases, however, it is sufficient to transfer only a part of this information depending on the purpose of the transmission and the powerfulness of the transmitter and receiver.

The purpose of the interchange determines whether the *formatted*, *processable* or *formatted processible* form is used:

- *Formatted Form*: These transmission formats only allow true-to-the-original reproduction of documents at the receiver's site. In essence, the transmission encompasses the layout structure and formatted content.
- *Processible Form*: These transmission formats enable the receiver to process the documents received. In essence, the logical structure and processible content are transmitted, and possibly also the document class definition. For this reason, therefore, the layout must be generated before the document is reproduced.
- *Formatted Processible Form*: This transmission format allows the receiver to reproduce documents true to the original as well as to process documents. It encompasses all components of the document architecture.

3.1 Interchange between Editors of Different Levels

With regard to powerfulness, the ODA model encompasses the functionality of a wide range of editors extending from simple text editors to top-level editors for mixed text/image documents which format interactively and which are document-class controlled. Figure 5 shows the interrelationships between ODA and the editors of various degrees of sophistication.

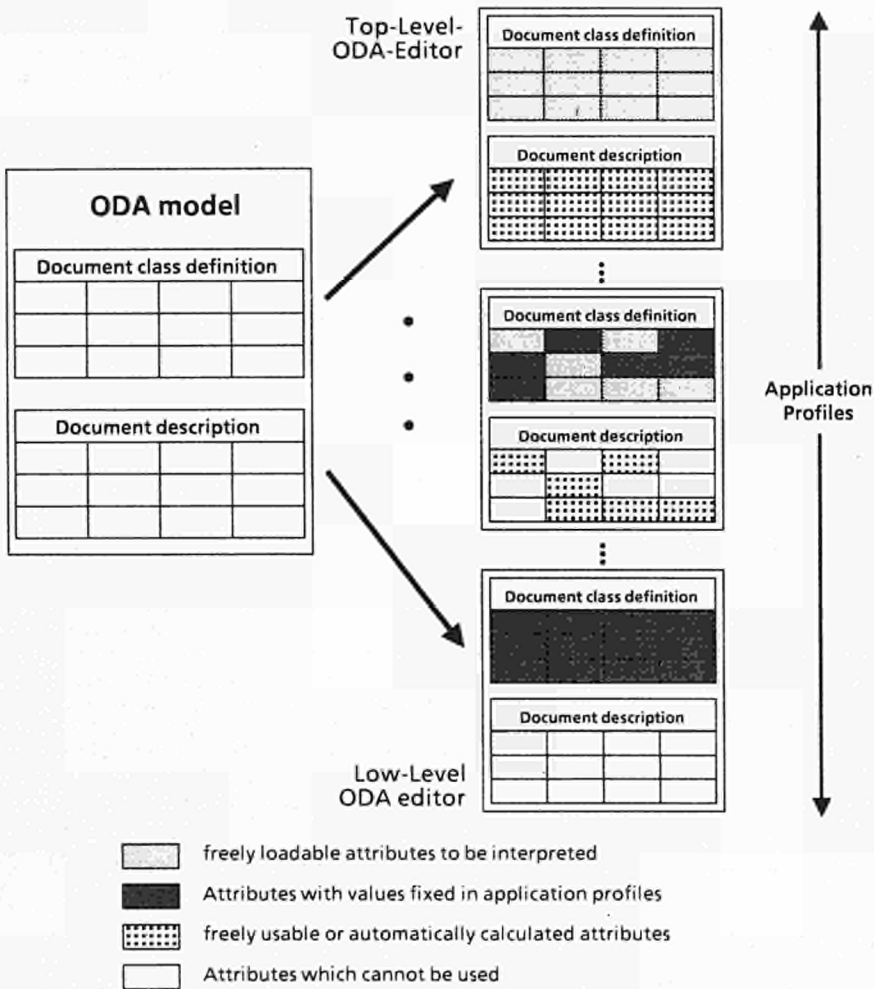


Figure 5: ODA and Editors of Different Levels

The ODA model fits to a general editor for mixed documents of different document classes. The set of document classes is defined by the power of the description mechanisms (for document class descriptions) as laid down in the standard. Together with a special document class description, ODA describes the functionality of a special editor for documents of this document class. The document description describes a document - not independently though, but rather as an instance of this document class and in the context of the ODA model. This division of the information into ODA

model, document class description and document description makes it possible to describe editors of different sophistication levels:

- The top-level ODA editor accepts every document class definition which can be described in ODA, i.e. all attributes in document class definitions can be loaded and are interpreted by the top-level ODA editor, which implies that the latter temporarily functions as a special editor for reports, business letters, order forms, etc.
- As regards the low-level ODA editor, ODA continues to apply, but the model is made more "stringent" by the fact that all attributes of the document class definition have been strictly prespecified. They are not interpreted during the run time but are taken into account during editor development.
- In the case of editors of medium sophistication classes, the ODA model is made more "stringent" by the fact that a number of the attributes of document class definitions have strictly fixed values. Furthermore, the value ranges of the freely loadable attributes to be interpreted can be restricted. In line with ODA, an editor of this type is document-class-driven, but the quantity of accepted document classes is restricted in comparison with the top-level editor.

A variety of interchange formats is conceivable within the ODIF interchange format space. However, a large number of different interchange formats would not promote the interchange of documents between open systems, since each system only accepts a few of these.

3.2 Family of ODIF Interchange Formats

The work begun in SPAG (*Standards Promotion Application Group*) has the purpose of defining such a family of ODIF interchange formats (*Document Application Profiles*). At the current time, four *Document Application Profiles* are being discussed (Figure 6).

These four *Document Application Profiles* (DTP = *Document Transfer Profile*) are upwards-compatible. The simplest of these is functionally compatible with teletex. The next level additionally allows all layout possibilities, but does not yet regard the document as an instance of a document class whose definition is also interchanged. The same applies for the third application profile which, in addition to *Character Content* also allows *Raster Graphics Content* and *Geometric Graphics Content* and is open for further *Content Architectures*. For these application profiles, ODA is made more stringent in the sense of Figure 10 due to the fact that *Document Superclasses* are defined in the application profiles, i.e. attribute values are permanently fixed in *Document Superclasses*, or their value range is restricted. The highest level allows to use full ODA, i.e. additional to DTP 3 each document is interchanged together with its document class definition.

4. IMPLEMENTATION OF AN ODA EDITOR PROTOTYPE

As part of the two ESPRIT projects HERODE (Handling the Electronic Representation of Mixed Text/Image Office Documents based on ECMA Standard 101 - Project 121) /10, 11/ and PODA (Piloting of ODA -Project 1024), a prototype of a top-level ODA editor was implemented and is shown at this ESPRIT Technical Week and an editor for document class definitions is in the process of being realized. The companies involved in the HERODE project are Siemens (D) and TITN (F), while Siemens (D), TITN (F), Bull (F), ICL (GB) and Olivetti (I) are involved in the PODA project.

As Figure 7 shows, the architecture of the ODA editor is shaped by ODA: There are two structure editors which adapt themselves to document classes using attributed grammars. Corresponding content editors are available for the various content architectures. Refer to /12, 13/ for more detailed information on the HERODE ODA editor. The user communicates with all these components via a common user

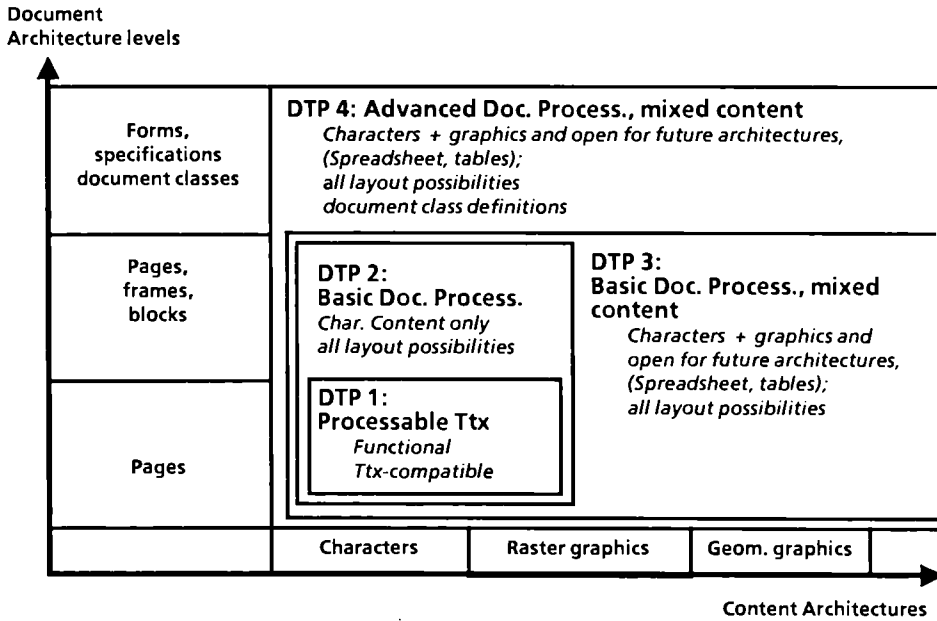


Figure 6: Family of ODIF Interchange Formats

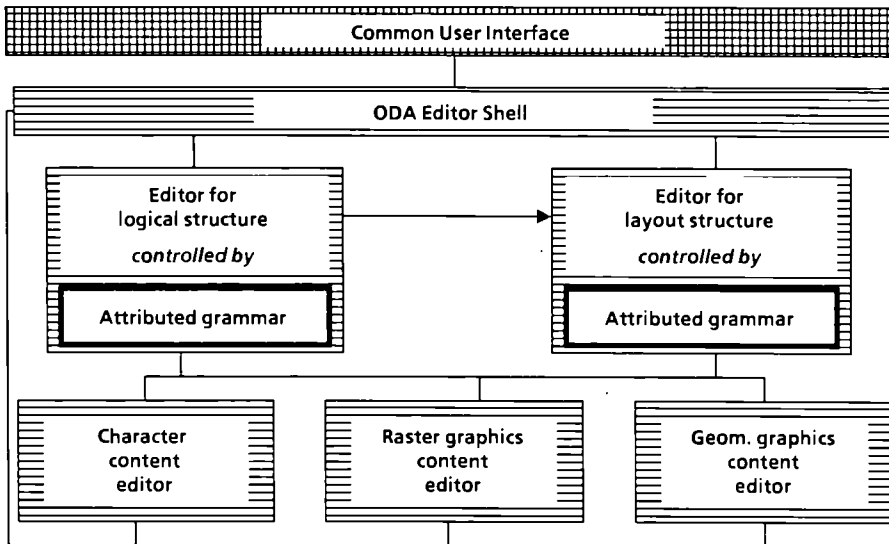


Figure 7: Architecture of the ODA editor

interface which gives the user the impression of a single, very powerful tool. Located between the common user interface (which is independent of application) and the individual editor components is the editor shells level.

5. STATE OF THE STANDARDIZATION AND PROSPECTS FOR THE FUTURE

ODA/ODIF has been described in ECMA Standard 101 from 1985 /14/. However, ODA/ODIF has since been improved and expanded considerably. The current status is described in Draft International Standard ISO DIS 8613/1-8. At the start of 1987, numerous comments and proposals for improvements to this DIS were submitted by the national standardization bodies and are currently in the process of being incorporated into the standard. It is planned that this improved version will then become an international ISO standard in November 1987.

This basic standard provides the foundation for other standards and recommendations which describe how ODA can be applied. CCITT COM VIII is working on the T.400 series (*Document Transfer and Manipulation*). T.411 to T.418 have the same wording as ISO 8613 and will most probably become a recommendation during this study period (up till 1988). T.4y encompasses services and protocols for noninteractive operations such as document transfer and interactive operations such as altering, generating or deleting parts of a document. This part will probably only become a draft recommendation in this CCITT study period. The same applies to T.4z, which describes additional concepts which allow *remote access* to and *remote manipulation* of documents, and consequently interactive operations.

SPAG will also introduce the *Document Application Profiles* into CCITT COM VIII. DTP 1 is intended to become a recommendation in this study period (up till 1988). DTP 2 and DTP 3 will be introduced as a draft for interim processing (1988-1990). CCITT is planning to draw up a recommendation for *Application Profiles* in the next study period (up to 1992). This is intended to include, among other things, the *Document Application Profiles* DTP 1 to DTP 4 and formats for videotex, teletex and fax derived from ODA.

In the next few years, ODA will be expanded in accordance with international ISO standard ISO 8613 to be upwards compatible. The following expansions are planned:

- Further *Content Architectures*, e.g. for language, mathematical formulae and chemical formulae
- Color in documents
- Notes on document sections
- Improved and expanded layout possibilities
- Security aspects
- Cross-references

As already mentioned at the start of this paper, the current ODA architecture model grew up with the goal of supporting the editing and formatting of documents. It is now important to expand ODA further so that it can also support the use of documents interchanged with ODIF in automated office procedures /15, 16, 17/. This means that it must also be possible to interpret the contents of a document as data, and to make dynamic links between documents and applications.

The already in ECMA, ISO and CCITT initiated discussion of these expansions shows that ODA, because of its clean subdivision into structures and content and its object-oriented approach, represents a model which is open for any necessary changes of this type, i.e. that any future requirements from the office world, which will certainly emerge but are not yet known, can be incorporated into further upwards-compatible developments of ODA.

References

- /1/ Lum, V.Y.: Automating Business Procedures with Form Processing. Office Information Systems (Proceedings, ed. N. Naffah), INRIA/North-Holland Publishing Company, 1982, pp. 7-38
- /2/ Donner, H.: Normen schaffen Freiheit im Buero, com (Siemens-Magazin fuer Computer & Communications), 4/85, pp. 8-13
- /3/ ISO/DIS 8613: Information Processing - Text and Office Systems - Document Structures, June 1986
 Part 1: General introduction
 Part 2: Office document architecture
 Part 3: Document layout and imaging process
 Part 4: Document profile
 Part 5: Office document interchange format
 Part 6: Character content architectures
 Part 7: Raster Graphics Content Architecture
 Part 8: Geometric Graphics Content Architecture
- /4/ Horak, W.; Krönert, G.: Techniques for Preparing and Inter-changing Mixed Text-Image Documents at Multifunctional Workstations. Siemens Forsch.- u. Entwickl.-Ber. 12 (1983), pp. 61-69.
- /5/ Furuta, R.; Scofield, J.; Shaw, A.: Document Formatting Systems: Surveys, Concepts and Issues. Computing Surveys, Vol. 14, 1982, pp. 417-472
- /6/ Meyrowitz, N.; van Dam, A.: Interactive Editing Systems: Part I and Part II. Computing Surveys, Vol. 14, 1982
- /7/ Horak, W.: Office Document Architecture and Office Document Interchange Formats.: Current Status of International Standardization, IEEE Computer, October 1985, pp. 50-60
- /8/ Krönert, G.: Normierung von Dokumentbeschreibungen, Handbuch der modernen Datenverarbeitung, Vol. 133, 1987, 47-54
- /9/ Chamberlin, D. et al.: JANUS: An interactive document formatter based on declarative tags. IBM Systems Journal, Volume 21, No. 3, 1982, pp. 250-271
- /10/ Horak, W.; Tartanson, F.; Colouris, G.: Handling of mixed text/image/voice documents based on a standardised office document architecture, ESPRIT 84 Conference, Brussels, North-Holland, pp. 395-410
- /11/ Krönert, G.; Lauber, G.; Loerscher, J.; Meynieux, E.; Postl, W.; Schneider, U.; Seisen, S.; Tombre, K.: Document editing and entry based on the standardized office document architecture. ESPRIT'86: RESULTS AND ACHIEVEMENTS, North-Holland, 1987, pp. 107-126
- /12/ Horak, W.; Krönert, G.: An interactively formatting document editor based on the standardised office document architecture, Proceedings of the IFIP conference on "Methods and tools for office systems", PISA, October 1986, pp. 287-300
- /13/ Krönert, G.: Objektorientierte Implementierung eines inter-aktiv formatierenden Dokumenteneditors auf der Basis des Standards ECMA 101, Tagungsband der GI-Jahrestagung 1986, Berlin, October 1986, 91-103
- /14/ ECMA: Standard ECMA 101, Office Document Architecture, September 1985
- /15/ Borron, H.J.: The concept of type: Considerations for Document Preparation, Retrieval and Communication. Office Information Systems (Proceedings, ed. N. Naffah), INRIA/North Holland Publishing Company, 1982, pp. 583-592 321-352 and 353-415
- /16/ Suda, P.: Some Basic Aspects of Office Specification Language Design. Office Information Systems (Proceedings, ed. N. Naffah), INRIA/North-Holland Publishing Company, 1982, pp. 521-527
- /17/ Tschritzis, D.: Form Management. Communications of the ACM, Vol. 25, No. 7, July 1982, pp. 453-478

THE ESPRIT PODA DEMONSTRATION OF ODA AT CeBIT 87

P.J.Robinson, ICL, EDITOR.
E.Koether, Siemens. I.R.Campbell-Grant, ICL.
O.R.Morel, Bull MTS. F.Fasano, Olivetti.

The ESPRIT PODA project partners of Bull MTS, ICL, Olivetti and Siemens demonstrated at CeBIT 87 that, using ODA, documents could be interchanged between quite different word processor products from four major European office automation vendors.

The ODA application profile, defined by PODA for CeBIT 87, has been adopted as the basis for a European profile and is helping Europe to lead in the harmonisation with non-European profiles.

A generalised ODA software architecture and tools were designed by the project. These will reduce the resources required to implement the multiple format conversions needed in multi-vendor systems. An initial implementation was completed and formed the key element of the CeBIT 87 demonstration.

The software components developed by the project according to the architecture are: company specific proprietary format conversion software; a portable ODA Storage Manager supporting a data structure interface "SODA" (providing random access to storage structures representing ODA constituents); portable software for converting ODIF to SODA and vice versa.

The PODA project partners are formative in the rapid acceptance and advancement of the ODA standard and intend to maintain their European leadership. The next objective for PODA is to demonstrate the interchange of documents containing character text, raster and geometric graphics using X.400 at CeBIT 88.

1. INTRODUCTION

The objectives of ESPRIT project 1024, Piloting of the Office Document Architecture (PODA) are described in terms of the new ISO and CCITT information interchange standard, the Office Document Architecture and Interchange Format (ODA). The objectives of the project are:

- i) to evaluate the suitability of the ODA standard as the basis for office systems;
- ii) to advance the ODA standard by submitting contributions to standards bodies based on the experience gained from prototype developments in new areas.

The partners in the PODA project are:

Bull MTS	France	(contractor)
ICL	United Kingdom	(contractor)
Olivetti	Italy	(contractor)
Siemens	West Germany	(prime contractor)
TITN	France	(contractor)
QMC IRL	United Kingdom	(sub-contractor)
SEPT	France	(sub-contractor)

This paper is concerned with one aspect of the first of the project objectives namely, to evaluate the suitability of ODA for interchange of documents between existing word processors.

For this aspect of the PODA project Bull, ICL, Olivetti and Siemens, recognised the need to validate as early as possible the ODA Standard by its use in a working system since ODA had been designed as a new solution to known problems and was not based on existing implementations.

The practical realisation of Open Systems Interconnection demands close cooperation between Information Technology vendors who traditionally compete in the same market. Such collaboration is expensive and benefits are not clear until after standards have been established commercially.

The PODA project partners recognised the need to rapidly construct prototype implementations which used ODA as the interchange format between existing word processors. The project also recognised that establishing the credibility of the new ODA standard would best be achieved by a public demonstration of its use. This was the reason for the demonstrations of ODA interworking at CeBIT 87 in Hanover and at Hanover 88. For credibility this demonstration was required to be between a number of quite different word processors from a number of vendors and was required to be achieved without modification to these products.

The ESPRIT programme has been the catalyst by which the European companies have made the investment in innovative research needed to lead this important area of Information Technology.

2. OFFICE DOCUMENT ARCHITECTURE AND INTERCHANGE FORMAT

2.1. The ODA Standard

The Office Document Architecture and Interchange Format, ODA, is a new information interchange standard for Open System Interconnection. ODA was initially formulated in Europe by the European Computer Manufacturers Association (ECMA) as the standard, ECMA-101 [1].

Currently, ODA is progressing through the ISO and CCITT ratification processes. In ISO it is at Draft International Standard status and in CCITT there is a series of draft Recommendations, these two specifications being precisely aligned. ODA is planned to reach full International Standard status as ISO IS 8613 [2] in early 1988 and become a series CCITT Recommendations as the T.410 series [3] later in the same year.

The ODA Standard enables documents containing character text, raster graphic (facsimile) images and geometric graphics to be encoded and interchanged electronically.

ODA has been specified to enable the interchange of electronic documents so that they can be reproduced as intended by the originator or they can be consistently edited by the recipient according to the originators intentions.

In order to achieve these objectives ODA specifies an abstract architecture for the representation of the information contained in a document. The ODA abstract architecture is a good technical basis for office applications and has been an interesting factor in several ESPRIT projects. For example one subject area of the HERODE project is to produce a syntax driven editor. This is complementary to other areas of PODA and is being used to develop a Portable European Document Handler in order to more fully evaluate the ODA standard.

2.2. ODA Application Profiles

The ODA standard, recognising the complexity of ODA, specifies how to form subsets of the total set of features and facilities for use by implementations servicing different levels of user requirements. These subsets are termed "Application Profiles" [4].

Initially the standards organisations intended to define application profiles. However, work by the PODA project found that application profiles were not sufficient to describe how existing equipment could support and conform to ODA.

Special technical considerations are required for the use of ODA for interworking between products designed independently of ODA. Whereas the products make their own particular style of realisation for any features such as page headings and footnotes, ODA has a powerful and generalised architecture for describing the properties underlying those features.

To solve this problem the concept of document "superclasses" was conceived by PODA. A superclass is the term used to specify how, for an ODA application profile, the users' perceived features, such as those mentioned above, are encoded in ODA using attribute values.

The concept of a superclass has been adopted by the various standards bodies defining functional profiles for ODA - first the European Standards and Applications Group (SPAG) and then CCITT, and Japanese and American standards bodies. As a result the superclass concept has been incorporated into the ISO/CCITT rules for the definition of application profiles for the use of ODA.

2.3. The PODA H'87 Application Profile

The application profile and superclasses defined by PODA for CeBIT 87 is called "H'87" [5]. H'87 has been adopted as the basis for a European profile for existing character mode word processors. The representatives from the PODA companies are also ensuring that non-European profiles will harmonise with this definition.

The document features that can be represented in the H'87 ODA application profile are representative of those likely to be found in documents containing character information only, ranging from very simple memos to highly structured reports. Figure 1 illustrates a representative page of a document that can be encoded in the H'87 application profile. The features that can be represented are as follows:

Logical features

- Sections with optional automatic section numbers
- Paragraphs
- Footnotes
- Footnote references (automatic and user defined)

Layout features

- Page formats
- Page numbering
- line spacing

Text Layout control

- Single column text
- Margins
- Tabulation stops
- Alignment
- Word wrapped within margins
- Non-breaking spaces
- Forced page breaks
- Soft hyphens

Text Rendition

- Bold
- Underline
- Subscript and superscript

Character Set

The graphic character set supported is a subset of the Teletex subreper-
toire of ISO 6937/2

BIOS Company Confidential Ref : WPED
Issue no : 3

Page layout → 1.3.2 This page illustrates some of the layout constructions that are commonly used in character text composition. To communicate this page so that it can be edited correctly by another word processor requires :

- a) Both systems support the same standard character sets to represent visible characters and control codes for printing;
- b) A standard representation of the additional information needed to distinguish features such as:
 - Headers, footers and footnotes;
 - Ruler lines, tables and indented paragraphs;
 - Justified text and centered text;
 - Pages numbers and section numbers.

Numbered sections → 1.3.3 Standard Character set

Compatibility of special characters such as currency symbols can be a problem, for example:

Non-breaking space → "The quotation from the vendor's representative, Mister S T Mayer was for £520 per unit"

The table below illustrates some of the effects of various control codes

Control function	Example of usage	Typical result of failure to edit correctly
Underlined characters	headings	Generally unacceptable documents presentation
non-break spaces	personal names	Initials and surname may appear on different lines
emboldened characters	emphasised phrases	Some loss of the meaning to the reader

Table 2 : Examples of character set controls

Centred text →

1.3.4 Section numbers

1.3.4.1 Many word processors can automatically generate section numbers.

1.3.4.2 When editing, if a user inserts a paragraph, subsequent paragraphs are automatically renumbered.

Page 2

Automatic page numbering →

Figure 1. Page of text in H'87

3. THE PODA ODA CONVERSION SOFTWARE ARCHITECTURE

3.1. Overall System Design

The system created by the PODA project for demonstrating the interchange of documents using ODA is presented in the overall system diagram of figure 2.

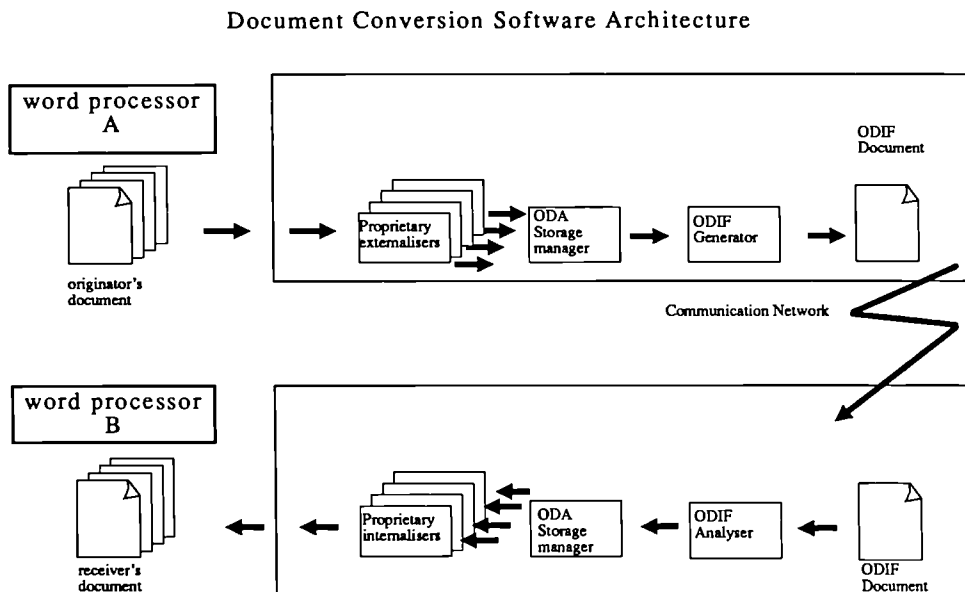


Figure 2. Overall System Diagram

The software architecture consists of, starting with a document to be processed in the format for word processor A and with the converted document in the format for word processor B, the following components:

- Proprietary Format Externaliser;
- ODA Storage Manager;
- ODIF Generator;
- ODIF Analyser;
- Proprietary Format Internaliser;

Of these software components the ODA Storage Manager, the ODIF Generator and the ODIF Analyser were all written in the 'C' programming language and were ported to each of the partners machines and operating systems.

A Proprietary Format Externaliser and a Proprietary Format Internaliser was developed by each of the four PODA partners, one for each of the company word processor formats used in the Hanover demonstration.

In addition to these major software components other components, such as utilities to print the stored ODA document in human readable form (see [6]), were developed to aid in testing the system.

3.2. The Proprietary Format Externalisers

The Externaliser software is responsible for examining the word processor document, to determine which user perceived features (as defined by H'87) it uses. The externaliser then proceeds to represent these features in the appropriate ODA constructs, again as defined by H'87. The component represents the required ODA constructs in an in-store, random access form of ODA. This random access form of ODA has been termed "SODA" [7] by the project, as the acronym for "Stored ODA".

The externaliser software component was made by all of the project partners, one for each of their chosen word processors used in the Hanover demonstration.

3.3. SODA and the ODA Storage Manager

The SODA form is created and accessed through the ODA Storage Manager. The ODA Storage Manager [8] is a set of library functions for performing and managing access to ODA constituents in their stored form and simplifies certain of the operations required on the more complex types.

The document interchange format defined by the ODA standard is termed "ODIF", as the acronym for "ODA Interchange Format". ODIF has been specifically designed for the interchange of documents over communications networks. ODIF uses the ISO Abstract Syntax Notation 1 (ASN.1) [9], which is developed from the CCITT Recommendation on Transfer Syntax [10]) (the CCITT 1988 version of this Recommendation is expected to be fully aligned with ASN.1).

ODIF is a serial stream format and as such is unsuitable for the processing required for conversion to and from existing document formats. SODA has been specifically designed for such processes as document format conversion where random access to ODA constituents is of prime importance.

SODA is a data interface which defines 'C' programming language structures representing ODA constituents. These include structure members used by the interface access mechanism and structure members corresponding to, or related to, all valid attributes in H'87 for each type of ODA constituent.

The ODA Storage Manager was designed by ICL and was ported to both Unix and MS-DOS systems and used by the other project partners in their components of the system. The ODA storage format, SODA, was also designed by ICL.

3.4. The ODIF Generator

The ODIF generator reads the document in its stored SODA format and converts it to ODIF. This conversion process is facilitated by the design of SODA which closely matches the ODA architecture.

The ODIF Generator was designed and implemented by Olivetti and subsequently ported by each of the project partners to their particular machine.

3.5. The ODIF Analyser

This component performs the inverse operation to that of the ODIF Generator: it reads a file of ODIF and converts it to the SODA form making use of the Storage Manager functions. The analyser also performs the essential check that the data stream is syntactically correct ODIF, a feature of great assistance during the development process.

The ODIF Analyser was designed and implemented by Olivetti and subsequently ported by each of the project partners to their particular machine.

The full details of the ODIF Generator and Analyser are described in [11].

3.6. The Proprietary Format Internaliser

This component processes the SODA form of a document and converts it to the format required for the appropriate word processor.

This component was constructed by each of the project partners, one for their chosen word processor.

4. THE CEBIT 87 HANOVER DEMONSTRATION

4.1. The Hardware and Operating Systems

The hardware, operating systems and word processors used by each of the partners at the Hanover demonstration are itemised in figure 3. These can be summarised as:

- two Unix based systems (from ICL and Siemens), a proprietary operating system (from Bull) and an IBM compatible MS-DOS system (from Olivetti);
- four proprietary word processors.

Company	Hardware	Operating System	Word Processor
Bull	QUESTAR-400	STARSYS	STAR PUBLISHING
ICL	CLAN3	UniPlus	OFFICEPOWER
Olivetti	PC M24	MS-DOS	OLIWRITE
Siemens	PC-MX2	SINIX	SITEX

Figure 3. CeBIT 87 System Components

4.2. Communications Facilities

The ODA standard has been specifically designed to be independent of any particular communications protocol. ODA is designed to be compatible with other OSI standards by encoding the interchange format ODIF according to ASN.1. ASN.1 is also the encoding standard used by the electronic mail standard - CCITT X.400. It is anticipated that in the majority of office systems ODA will be used in conjunction with the X.400 mail standard as the communications facility.

At CeBIT 87 however, the prime objective of the demonstration was to show ODA being used as the interchange format between existing word processor products and not the communications facilities, hence a simple communications protocol was used. The chosen protocol, common to all the systems, was "Kermit" running over RS - 232 links between the machines.

5. FUTURE PLANS - CeBIT 88

The PODA project has already commenced defining the next public demonstration to be held at CeBIT 88. The project plans to build on the successful ODA conversion system developed for CeBIT 87 and to define a new ODA application profile, H'88.

5.1. The H'88 ODA Application Profile

The H'88 application profile will add to the features established for H'87. Most importantly the H'88 profile intends to include all content types defined by the ODA standard. These are character text, raster and geometric graphics. The work on the H'88 profile is being submitted as the basis for a European application profile for the interchange of documents requiring this level of facility.

5.2. Office System Software

The intention is to include in the demonstration word processors, or document processors that can include character, raster and geometric content.

5.3. Communication Facilities

For the Hanover 88 demonstration it is proposed to move on from the elementary protocol of CeBIT 87. It is proposed to use the CCITT X.400 Electronic Mail facilities of each of the partners as the mechanism for communicating the documents between systems. Demonstrating ODA together with X.400 will show the two key-stone standards of Office Automation in use.

6. RESULTS

The ESPRIT programme has enabled, through the PODA project, traditionally rival European information technology companies to cooperate in the important area of establishing standards for the interchange of documents.

The results of this task of the PODA project have exceeded expectations. Not only has a standard, first ratified in Europe (ODA) been used to publicly demonstrate the electronic interchange of documents but also important software components have been developed. The ODA Storage Format and Storage Manager have general applicability for future ODA systems. The ODIF Generator is also of general use to future systems. The ODIF Analyser is a key element in the important area of ODA conformance testing.

In the area of standards, the recognition of the need for ODA application profiles to include a document superclass definition is demonstrated by the inclusion of this concept in the ISO and CCITT standards by those standards organisations.

7. CONCLUSION

The ESPRIT PODA project publicly demonstrated at CeBIT 87 that, using ODA, documents could be interchanged between the existing word processors of four major European Office Automation vendors.

Each of the four companies provided conversion software between its proprietary format and ODIF the ODA Interchange Format. Documents could be sent, in ODIF, from one word processor to that of any of the other companies where it could in turn be edited and forwarded to another.

The PODA project partners have gained valuable practical experience in the design of systems to process documents encoded in the new standard for document interchange, ODA.

This part of the PODA project has developed several important software components for future ODA systems, these being SODA, the ODA Storage Manager the ODIF Analysers and the ODIF Generator.

The project has also been able to make significant contributions to the specification of an ODA application profile for use with existing word processors, by its submissions to the European Standards Promotion and Application Group.

The project aims to build on the knowledge gained by the existing developments to include so called "mixed-mode" documents with character, raster and geometric graphic contents. The project feels confident that at CeBIT 88, it can establish another milestone in the development of systems using the ODA standard and demonstrate how ODA and European expertise can help fulfil the long awaited promises of Office Automation.

ACKNOWLEDGEMENTS

The editor would like to acknowledge the help given in preparing this paper from colleagues in the ESPRIT PODA project team and from those in ICLs Advanced Products Sector. Without their contributions, cooperation and criticism this paper could have not been prepared.

REFERENCES

- [1] Standard ECMA-101, Office Document Architecture, September 1985.
- [2] Office Document Architecture (ODA) and Interchange Format, ISO DIS 8613, June 1987.
- [3] Draft CCITT Recommendation T.410 series, Document Transfer, Access and Manipulation (DTAM), July 1987, (pending publication).
- [4] Office Document Architecture (ODA) and Interchange Format, ISO DIS 8613-1.2, clause 9: Document application profiles, June 1987.
- [5] Pennell, R.F. and Guerra L., ODA H'87 Profile Definition, ESPRIT project 1024 deliverable, December 1987.
- [6] Office Document Architecture (ODA) and Interchange Format, ISO DIS 8613-2.2, Annex B.2, Table B.1, June 1987.
- [7] Pennell, R.F., Stored ODA Interface (SODA), ESPRIT project 1024 deliverable, December 1986.
- [8] Feather S., Office Document Architecture Storage Manager structural design, ESPRIT project 1024 deliverable (pending publication), June 1987.
- [9] ISO 8824, Information processing - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), pending publication.
- [10] CCITT Recommendation X.409, Presentation Transfer Syntax and Notation (Malaga-Torremolinos, 1984).
- [11] D'Amato, T. and Guerra L., ODIFSM: Internalisation and Externalisation of an ODIF stream, ESPRIT project 1024 deliverable, June 1987.

Retrieval of Multi-Media Document Images in MULTOS

P. Conti (**), F. Rabitti(*)

(*) Istituto di Elaborazione della Informazione
Consiglio Nazionale delle Ricerche
Via S.Maria 46, Pisa (Italy)

—
(**) Olivetti D.O.R.
Via Palestro 30, Pisa (Italy)

ABSTRACT

In this paper we address the problem of retrieving images from large databases of multi-media documents, containing images as components. The queries are specified giving a partial description of the image content. The image analysis process, which is specific for an application domain described in advance to the system, is outlined in the various phases. Images, both pictorial and graphical, are represented using a structural approach and are interpreted according to the theory of evidence. The retrieval process is executed on the access structures generated by the image analysis process.

KEYWORDS: *Low level image analysis, High level image analysis, Image understanding, Attributed Relational Graphs, Pictorial images, Graphical images, Image Storage, Image Retrieval, Image Query Language.*

1. Introduction: the problem of image retrieval

Office documents are a particular class of objects that have complex structures and contain different data types: text, image and voice. Systems for the storage and retrieval of this kind of documents are under study, but they make little use of the information contained in their image and voice parts. In general, their approach to the retrieval problem is to link images and voice comments to other more structured document components, such as the corresponding captions or annotations, and then exploit this last information to access the corresponding documents. In other words images and voice are not used as active sources of information. Within the MULTOS [Cons86] project our goal is instead to define, implement and test a system that allows the classification and retrieval of multimedia documents exploiting the content of each type of data that they contain.

In this paper we will focus on the role of the images in a document and in particular, to facilitate the explanation of the approach we propose, we will consider images as entire documents.

The problem of image retrieval has received a particular attention in the context of pictorial data base systems. These systems have been proposed in order to handle the growing amount of electronically stored digital images and a key function of these systems is image retrieval: suitable query languages and access structures for images must be implemented [Chan81].

Most of these systems are application specific, that is, the manner in which images are stored, organized and retrieved is specific of a certain application and cannot be generalized to different applications. In many systems, images are stored in files which are linked to other image files into structures for image retrieval and presentation according to the logic of the application [Hero80].

Other systems are based on an underlying database whose schema describes the image content and composition [Econ83]. These systems can exploit the flexibility of a Data Base Management System (DBMS). In particular, the query language and the access structures implemented in the DBMS are very powerful for retrieval operations. But they are limited by the fact that the content of the images must be described using data models which were developed for database systems rather than image systems, and so they lack the expressive power needed for images. In fact, images are inherently different from database records: these records can be divided in different classes according to their interpretation. The record structure (i.e. the schema) can be described at class (i.e. type) level. Since the ratio of instances per type is very high for database systems, the resulting storage structures and access methods are very efficient. On the contrary, each image may have

its own particular structure, and a whole semantic network [Tsic82] may be necessary to completely describe each image instance.

A new application area where the problem of image storing and retrieval has been addressed is office automation. A new information object is defined, the multimedia document, where images are combined with attribute data, text, and voice [Tsic83]. Systems for the storage and retrieval of large volumes of these documents are under study. One of the main functions of these systems is the access to multimedia documents based on their content. However, while these systems incorporate efficient access methods for attribute data and text, they can do very little for images, as well as voice. As just mentioned before, their approach is often to link images to other more structured components of the multimedia document, and then exploit the combined access to them [Tsic83]. (This is the approach followed in the design and implementation of the first prototype of MULTOS [Bert85].)

For the retrieval of documents by content, multimedia document systems can exploit very efficient techniques borrowed from DBMS [Ullm82], when formatted data is concerned, or from Information Retrieval Systems (IRS) [Salt83], when text is concerned. Comparable techniques are not available for images.

The main conceptual problem in dealing with images derives from the difficulty to exactly define and interpret the content of images. Images can be very rich in semantics, but are subject to different interpretations according to the human perspective or the application domain. On one hand, it is difficult to recognize the objects (with the associated interpretation) contained in an image, on the other hand is difficult to determine and represent the mutual relationships among these objects, since they form structures which vary greatly from image to image.

One approach could be to face directly the problem of image understanding. This implies the combined use of sophisticated pattern recognition techniques (usually expensive in terms of computing power and special hardware required) and advanced artificial intelligence techniques to represent and organize the knowledge expressed in the images [Hans78]. In this case, access structures could be built in terms of the image representations in the image knowledge base. This class of expert systems is still in an experimental phase, limited to some application specific prototypes [Flee84], whose results cannot be generalized.

Another approach could be to apply DBMS or IRS techniques. However, with respect to DBMS's, it is difficult to recognize regular structures of objects contained in images, and then organize image instances into a limited number of types, to which the interpretation is associated. This is the approach required by the strictly typed data models adopted in

database systems [Tsic82]. In IRS, instead, a free formatting of text is allowed, usually respecting some loose hierarchical structuring in sections, sub-sections, paragraphs and sentences. These systems do not attempt to understand the text (unless some expert system approach is adopted), but still allow an effective retrieval on text. In fact, as opposed to image objects, they can exactly recognize words (as ASCII patterns), on which they base their retrieval capabilities, with the possible help of thesauri to support synonyms [Salt83]. This is possible, in case of text, because a common semantic is associated to the words used in the natural language. Hence, both DBMS and IRS approaches cannot be directly applied to the image retrieval.

In addressing the problem of image retrieval on large volumes of stored images, if we want to think of a system doing for images what DBMS and IRS do for formatted data and text, we must accept some indeterminateness, characteristic of images, and then deal with the inaccuracy introduced by this fact. We have recognized the exigency of adopting, at some level of the image recognition process, a non-boolean logic allowing some form of approximate reasoning (in [Rabi87] the fuzzy logic approach is explored).

Thus, we have decided to investigate an approach based on the mathematical theory of evidence, for the image recognition and description part, and on the probabilistic theory, for the retrieval of the images previously analyzed. This combined approach allows the representation of different degrees of similarity between objects, the recognition of images, and contained objects, with certain belief, and the definition of classes of images/objects with unsharp boundaries.

However, a major problem in handling raw images is the "element recognition". The first step in image analysis is infact the decomposition of the images into relevant and identifiable elements, which are the basic components, constituting the building blocks of the image structure. During this step, the image is partitioned in relevant regions, corresponding to the elements, that are matched with the model elements, in order to be recognized together with their relative position.

In this paper, an approach is presented for the analysis and retrieval both of raw and graphical images. It is required that images belong to a specific, or more then one, application domain described in advance to the system.

We must stress the point that the real goal of this image analysis process is not to attempt any deep image understanding, but is to support the image retrieval process. In fact, the system allows the user to query the images, already analyzed and stored, giving some specification of their content. The image query processing is based on special access structures (i.e. image indices) which are generated when the image analysis is performed.

2. Image Classification and Retrieval in MULTOS

It has been shown that the major difficulty, related to image retrieval by content, is to define and interpret the content of images given the large variety of objects that can be contained and the complexity of the relationships among them.

For these reasons, it is firstly necessary to choose a class of images to handle (i.e. application domain) and then to build an efficient image classification system that allows, after a representation and analysis phase, the description of images in terms of the objects they contain. Moreover, in order to facilitate and to speed up retrieval operations, an efficient image database is needed, in which image descriptions are organized and suitable access structures are built.

In the image sub-system of MULTOS we can identify four different tasks:

- 1) The purpose of this task is to describe the characteristics of the application domains of the images to be classified and retrieved. This description is in terms of patterns and rules that will be exploited in the image analysis phase (Task 2) to correctly "understand" the images entered into the system.
- 2) The purpose of this task is to analyze the images entered into the system using the definitional information (particular of the application domain) specified in Task 1. This process can be extremely complex and time consuming, mainly for images entered as bit-maps. For this reason we want to split this task in two sequential phases:
 - 2A) The low-level image analysis accepts an image constituted by a bit-map and recognizes the basic objects in it, their relative positions and the associated distinguishing attributes. Since the number of these basic elements, obtained from the image scanning process, can be very large in a single image, a very efficient approach is required. It is not possible to adopt a rule system, based on a generalized inference mechanism, since the computational complexity would be exponential. We need instead a polynomial computation complexity, even if we have to pay this with a description system less rich in semantic content. For this reason we have adopted an approach based on the Attributed Relational Graphs.
 - 2B) The high-level image analysis starts from the basic objects obtained in the previous phase and then applies a body of rules for the recursive recognition of the more complex objects contained in the image. In this phase a generalized inference mechanism is used. The computational complexity is acceptable now, since fewer higher level objects are present in the image. The result is the in-

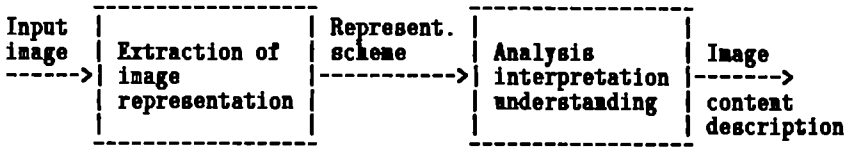


Fig. 1. Major Steps in Image Analysis

terpretation of the image, in terms of contained objects with associated belief and plausibility, and their attributes. The theory of evidence is used for the extraction of the belief intervals of the different objects interpretations.

- 3) The information obtained in Task 2 (i.e. interpretation of the images and information about the contained objects) is then used to generate access structures on image content, which can be used for efficient image retrieval (in Task 4). Access structures are mainly indices on the objects contained in all the image, with the associated attributes, and clusters about the interpretations of the images.
- 4) The purpose of this task is to accept queries on image content and to retrieve the images, stored in the system, which satisfy in some degree the query specification. For the image query processing task, the access structures generated in Task 3 are used.

3. Low-level Image Analysis

During the image analysis phase, there are two major steps performed by the MULTOS subsystem, as shown in Fig.1.

The first phase is concerned with the extraction of an adequate form of knowledge from the image data, representing it in some appropriate formalism.

In the second phase the actual analysis and processing of the image is performed, using the extracted representation. In order to improve the second phase of the scheme two central issues have to be considered:

- (a) the need of an efficient form of image representation (output of the first phase and input of the second one) on which performing the actual image analysis

(b) the support of a knowledge representation system for the interpretation and understanding steps.

As far as point (a) is concerned, several forms of image representation have been used in image analysis; traditionally there have been two major approaches to global image representation: the parametric approach and the syntactic approach.

- In the parametric approach, objects are represented by vectors of features (e.g. color, size, etc..), which can be easily measured from images.
- In the syntactic representation, the image features are represented as sets of symbolic entities, that is an alphabet of image primitives. The structural relations between these features are represented by mutual relations between symbols of the alphabet.

At the moment, in the field of image understanding, it seems particularly useful the reasoning with structural descriptions, in which objects are represented in terms of their parts and interrelationships between parts.

The corresponding powerful approach for image representation is obtained combining the parametric approach into the syntactic approach, in the sense that the semantic (image features and relationships values) is incorporated into the syntactic representation as structured attributes. The image features are represented by an alphabet of entities, where the attributes represent some semantic parameters of the structural features. The semantic information on the relationships among the image features is represented by the attributes associated with the relations between their corresponding entities.

Following this method, to adequately choose an alphabet of entities and their interrelationships, three central issues arise:

- (a) Which features should be extracted from an image in order to have a useful description of its physical properties and their relationships?
- (b) In which form should features be combined into object models so that the description is appropriate for recognizing all objects in the given image?
- (c) How should the correspondence or matching be done between image features and object models in order to recognize the parts in a complex image?

As just mentioned before, to answer these questions, and in particular the first one, it is necessary to work on a given class of images. Initially, in order to test the system, we will consider a class of images whose objects have simple geometric structures. The alphabet of entities used to describe these objects will be as simple as their basic geometrical components (i.e. line segments, arcs and closed curves).

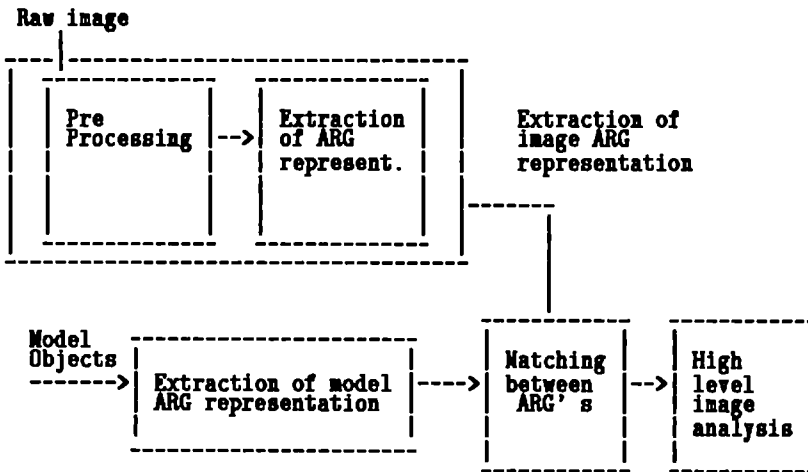


Fig. 2. Major Steps in Image Low Level Analysis

In the remainder, the general scheme of the low level analysis phase, performed by the proposed system, is shown. The last mentioned image representation approach, is utilized.

3.1. Attributed Relational Graphs

The proposed system is now described and its block diagram is shown in Fig.2. The system utilizes the Attributed Relational Graph (ARG) [Eahe86] for the knowledge representation and performs the analysis and understanding tasks on it.

An ARG is a relational structure which consists of a set of nodes and a set of branches representing the relations between the nodes. Both nodes and branches may have some attributes assigned to them. Usually, nodes are used to represent some objects or part of objects in the image, while their properties are assigned as attributes to the respective nodes. The relations between two objects are represented by attributed branches between the corresponding nodes.

The main operations related to each step of the above scheme are the following:

- During the preprocessing phase (that is not investigate within this paper) simple physical measurements (e.g. filtering, edge detection) are performed on the raw image

in order to produce a set of image primitives (i.e. object boundaries), input to the second step.

- During the ARG extraction, through a recursive process, the image information content is mapped from an input alphabet of local primitives (pixel values), into an output alphabet of image global features (the just mentioned structural alphabet of entities). These extracted entities are then correlated establishing the relationships between them to obtain the image ARG representation.
- During the last step of the low level analysis the system performs an inexact matching between the image ARG and the model object ARG's (evaluated as graph distance measure [Esh84]), in order to obtain the image description in terms of contained basic objects and to carry on the image understanding process.

This last step is very important in fact the most significant question in practical applications is not whether two images (subimages or objects) are identical, but rather how similar they are. Thus, an interesting issue for an image understanding system is in the actual possibility to define and calculate a distance measure between images. This distance measure, or better a tolerance specified in it, allows the system to deal with partially overlapped objects as well as noisy and distorted images.

For these reasons, graphs and in particular ARG's are a powerful tool for image representation, because of the possibility to perform the matching between images through the matching of their respective ARG's. (Calculating the graph-to-graph distance is a problem that has been studied extensively and can be performed with polynomial complexity). At the end of this step we have the image description in terms of the basic objects it contains.

4. High-level Image Analysis

If special image editors are used the analysis can start from this step. The high-level image analysis is performed in three steps:

STEP-1: Recursive Object Recognition

The purpose of this phase is to compose more complex objects from the derived through low-level image analysis basic ones. It is accomplished by a recursively applied production rules from a set defined for the correspondent application domain.

The rules define also the degree of recognition of an object as a distance between objects implied in the rule and those found in the image.

To define the distance, we call element either an object or a link between two objects.

The distance between two elements (either objects or branches), one in the structure contained in the image graph and the other in the rule structure, is given as follows:

$$Dist = \begin{cases} \frac{|Weight(graph\ element) - Weight(rule\ element)|}{Weight(rule\ element)}, & \text{if } Weight(rule\ element) \neq 0; \\ Weight(graph\ element), & \text{otherwise,} \end{cases}$$

where the weigh represents the parameters, associated to the objects and it is to be given in the domain description phase.

The recognition degree (*RD*) of an element is given as:

$$RD = \begin{cases} 1 - Dist, & \text{if } Dist \leq CUTOFF; \\ 0, & \text{otherwise,} \end{cases}$$

where *CUTOFF* is the minimal value (eg. 0.5) below which an element is not recognized at all. In this way elements with low recognition propabilities are eliminated.

If we replace one graph part with an element, using a rule, the *RD* of the new element is obtained as average value of *RD* of all the elements and branches in those graph part.

After this step a sequence in the form (1) is obtained:

$$(1) \quad [O_{1\ 1}(\mu_{1\ 1}), \dots, O_{1\ s_1}(\mu_{1\ s_1})], \dots, [O_{n\ 1}(\mu_{n\ 1}), \dots, O_{n\ s_n}(\mu_{n\ s_n})].$$

Such a sequence denotes an image with *n* distinct physical objects. The unit $O_i(\mu_{i\ j})$ is a semantical representation of the physical object *i* ($i = 1, 2, \dots, n$) in the image in the *j*-th ($j = 1, 2, \dots, s_j$) recognition (i.e. a semantic object). $\mu_{i\ j}$ is the degree of recognition of the *i*-th physical object in the *j*-th recognition.

In our approach we chose a logic programming language, namely Prolog, to express the object recognition rules. Thus, we used the Prolog's inference mechanism to perform the object recognition, as part of the image analysis function.

STEP-2: Image Interpretation

Let us suppose that a graph portion, corresponding to a physical object in the image, has been recognized through several rules as different semantic objects each with certain recognition degree.

Other representations of the physical object are obtained. They include the semantic objects, constructed from identical semantic objects by converting the different recognition

degrees into a belief interval. For this purpose the Dempster-Shafer theory of Evidence [Gord84] is applied.

The belief function $Bel(O_i)$ ($i = 1, 2, \dots, n$) gives the total amount of belief committed to the object O_i after all evidence bearing on O_i has been pooled. The function Bel provides additional information about O_i , namely $Bel(\bar{O}_i)$, the extent to which the evidence supports the negation of O_i , i.e. \bar{O}_i . The quantity $1 - Bel(\bar{O}_i)$ expresses the plausibility of O_i , i.e., the extent to which the evidence allows one to fail to doubt O_i . The interval

$$[Bel(O_i), 1 - Bel(\bar{O}_i)]$$

is called belief interval.

Furthermore, Barnett's scheme [Barn81] is used to compute the belief interval for every object interpretation in the image.

First, all recognition degrees of the identical interpretations of a physical object O_i ($i = 1, 2, \dots, n$) in the image are combined. If μ_1, \dots, μ_t represent different degrees of object recognition, the combined support to the object is

$$p_i = 1 - (1 - \mu_1)(1 - \mu_2) \dots (1 - \mu_t).$$

Then $Bel(O_i)$ and $Bel(\bar{O}_i)$ are calculated:

$$Bel(O_i) = K \times [p_i \prod_{j \neq i} d_j]$$

and

$$Bel(\bar{O}_i) = K \times \left(\left[\prod_j d_j \right] \left[\sum_{j \neq i} \frac{p_j}{d_j} \right] \right),$$

where:

$$K \times K^{-1} = 1$$

and

$$K^{-1} = \left[\prod_j d_j \right] \left[1 + \sum_j \frac{p_j}{d_j} \right],$$

$$d_i = 1 - p_i$$

and $j = 1, 2, \dots, n$.

After this step, the sequence (1) is reduced to the sequence:

$$(2) \quad [O_{1 \ 1}(Bel(O_{1 \ 1}), 1 - Bel(\bar{O}_{1 \ 1})), \dots, O_{1 \ q_1}(Bel(O_{1 \ q_1}), 1 - Bel(\bar{O}_{1 \ q_1}))], \dots,$$

$$[O_{n_1}(Bel(O_{n_1}), 1 - Bel(\overline{O}_{n_1})), \dots, O_{n_{q_n}}(Bel(O_{n_{q_n}}), 1 - Bel(\overline{O}_{n_{q_n}}))],$$

where $q_i \leq s_i (i = 1, 2, \dots, n)$.

In this sequence, objects with small belief are omitted.

STEP-3: Image Clustering

The interpretation of an image, as result of the analysis, is in terms of the interpretations of the composing objects. Therefore, the sequences (2) is an image interpretation.

The clustering process consists in adding to the obtained image description information about the membership degrees of the image to every defined class. This is made by comparing the image interpretation with all the given class descriptions (i.e. the class centroid images). This process is similar to the clustering process in Information Retrieval Systems [Salt83].

The membership degree of an image to a class is assumed to be the inverse of the distance of the image interpretation to the class representative image. The distance between two images is defined as the vectorial distance between their interpretations and is evaluated as a distance among object attributes. According to this, an image is represented by a vector whose elements are the objects, and the value of each element is the mean value of the belief interval of the corresponding object in the image interpretation. The vector elements should be ordered according to some global ordering of all the objects in the domain: if some object is not present in the image the element value is zero.

After this computation, the clustering description of the image is expressed as a sequence(3):

$$(3) \quad K_1(\mu_1), K_2(\mu_2), \dots, K_p(\mu_p),$$

where μ_i is the membership degree of the image to the class K_i .

5. The Complete Image Analysis Process

In this paragraph a complete view of the proposed image classification system is given.

1. The low-level image analysis is constituted by the following steps:
 - 1.1. The scanner performs the digitalization process on the input image (in case of pictorial images).
 - 1.2. The digitalized image is used to extract its ARG representation, while a copy of its is compressed and stored in an image database. That is necessary to maintain images in their "original" form.

- 1.3. When the image ARG representation is extracted it is stored in an ARG database. This database is very useful for a possible later improvement of the system. In particular, whenever it will be needed to extend the set of the basic objects, it will be possible to reclassify the already classified images, directly accessing their ARG's and matching them with the Basic- Object ARG's. In other words, the whole ARG extraction process must not be performed again. During the system initialization phase images represent basic objects and therefore their ARG's will be stored in an another database: the basic object database
 - 1.4. When the image ARG representation is extracted the next step is to match it with the basic object ARG's. The result of the matching is the image description, in terms of the basic objects it contains.
2. Using this information (about the contained objects and their relative position) it is possible to perform the high-level image analysis. We should notice that in case of graphical images, the low-level image analysis is much simpler of what has been previously described for pictorial images. The basic elements, described in terms of the editor graphics primitives, are organized in an ARG structure constituted by the basic objects of the application and the mutual relationships (i.e. their relative positions and the associated distinguishing attributes). This ARG structure hides the peculiarities of the graphical representation of the basic objects, generated by the graphical editors, and can be understood by the body of rules of the next phase.
- 2.1. **Recursive Object Recognition:** In this phase the system applies a body of rules describing the objects, which are semantically relevant in the application domain, in terms of their composition by other objects. The rules are recursively applied: the application of a rule transforms the object graph into another in which a set of objects has been substituted by a more complex object (described in the rule). This process is applied as long as possible, leading to the recognition of all the semantic objects in the image. Since different rules can be applied to the same structure of objects, these rules can recognize different semantic objects in the same object of the image. Moreover, the rules describe the recognition degree of each object recognized, according to how well the object structure investigated match with the rule specification.
 - 2.2. **Objects Interpretation:** The result of the previous phase is a set of objects recognized in the image with associated recognition degrees. Then the theory of evidence is used to generate all the different object interpretations, for each object identified in the image, with the associated values of belief and plausibility. Attributes to object interpretations can also be generated.

2.3. Image Interpretation and Clustering: The image interpretation is constituted by the sequence of all the interpretations of the objects found in the image. It is also possible to define the distance between two images in terms of their interpretations. The image clustering process is in principle similar to the document clustering of text documents used in Information Retrieval Systems [Salt83]. The most significant classes of images in the application domain are defined in terms of a representative image for each class. Then images interpretations are clustered comparing them with class representative images. Image clusters can then be used in the image retrieval process.

6. Storage and Retrieval of Images

We will now discuss the activities related to image compression, storage, indexing and retrieval.

6.1. Image Compression

The aim of image coding is to reduce the amount of information needed to specify an image, or at least an acceptable approximation to the image.

Compact encoding techniques allow one to store images in less memory space, or to transmit them in less time. In the above proposed system the used image compression technique is an Error-Free technique. In general, Error-Free techniques take advantage of the nonrandomness of images to obtain codes that are, on the average, more compact than the original images, while still permitting exact reconstruction of them. The compression technique, utilized by the system, is based on the idea that since the gray level in an image do not occur equally often, it is possible to compress the image by using short codes for the frequently occurring gray levels, and longer codes for the rare ones. (More details in [Rose82]).

6.2. Image Storage

In our approach, along with the file containing the image presentation we must also store the added information resulting from image analysis.

For pictorial images, the image file will contain the result of the compression of the original image, according to the technique previously discussed. For graphical images,

the image file will contain the sequence of the graphics primitives used for the image composition by the graphics editor. In general, they will depend on the particular editor used. In a system based on the GKS graphic standard [ANSI84], the image file will be constituted by the GKS metafile, stored as a set of segments, each containing graphic elements with the associated attributes.

6.3. Image Indexing

The complete image description, resulting from the analysis phase, is considered in terms of the probabilistic model as composition of objects, at different level of complexity, with the associated interval of belief. However, it is essential to find a suitable storage representation for this added information since the efficiency of the image retrieval process is based on it. For this purpose, it is useful to define some type of indexing on objects and associated belief intervals.

The image access information is stored in an "image header", associated to the image file. In this header we store:

- One sequence (3), containing the image clustering description. Each term of these sequence contains the membership degree of the complete image in one of the image classes of the application. This kind of information is more synthetical, since it refers to the image as a whole.
- one sequence (2). The same object may appear more times in the sequence, one for each appearance of that object in the image interpretation. This kind of information is more analytical, since it refers to the composition of the image.

Access structures (that is, the image indices) can be built for a fast access to image headers. Two type of indices are constructed:

- **Object index**

For each object a list is maintained. Each element of the list is constituted by a list of elements (BI, IMH), where IMH is a pointer to an image header, meaning that the object is present in that image, BI is the associated belief interval. For query processing, it is very important to maintain the list in decreasing order of BI. The order is computed using the mean values of the belief intervals.

- **Cluster index**

For each class defined in the application, a list of elements (MF,IMH) is maintained. IMH is a pointer to an image header, corresponding to an image with a non-null degree of membership to this cluster, and MF is the value of the membership degree.

For query processing, it is very important to maintain the list in decreasing order of MF.

6.4. Image retrieval

The user has the option between two types of interfaces for querying images: a query language interface [Ull82] and a query "by-example" interface [Zloo77].

According to the query language, the user specifies a query statement of the form:

RETRIEVE IMAGES <image_clause> .

The <image_clause> contains the <cluster_clause> and-or the <object.-clause>.

The <cluster_clause> is a list of <cluster_predicate>'s, each of the form:

<class_name> <similarity_degree> .

This predicate indicates that the images in the database should be retrieved with a similarity degree higher than the <membership_value> to at least one of the classes named in the list. The <cluster_clause> may be missing if the <object_clause> is present.

The <object_clause> is a boolean combination of <object_predicate>'s, each of the form:

<object_name> <degree_of_recognition>

The <object_clause> must be evaluated, according to the boolean expression, taking into account only the images in the database containing those objects, named as <object_name>, with the lower value of the belief interval higher than the <degree.-of_recognition>. The WITH operator, in a <object_predicate>, serves the purpose of adding conditions to the attributes associated to the object named as <object_name>.

All the stored images having a distance lower than the required accuracy will constitute the query answer set. With this approach, the query answers can be ordered by decreasing similarity to the query specification, so a user may limit the size of the answer and can receive a ranked output of the retrieved images (These advantages are typical of the information retrieval techniques [Salt83]).

A browsing facility becomes essential in this approach. That is, the user should have the possibility of browsing through the retrieved images. Since the image retrieval is not an exact process (since there is no exact way of defining the image content) and even the user may forget essential characteristics of the sought images, several non pertinent images can be retrieved as a result of a query. Moreover, relevance feed-back and query reformulation [Salt83] are emphasized, since at any moment the user can go back to the

query formulation step, if dissatisfied by the results which he is getting, and change some aspects of the query specification.

Using the "by-example" interface, the user queries the image database describing to the system a prototype of the wanted image. In the prototype creation the user should put all his recollection about the objects contained in the target images. As part of this query, the user should define the degree of similarity he wants between the prototype image and the retrieved images. This accuracy parameter could then be translated in terms of probabilistic distance.

We do not elaborate here on a possible definition of an image query by-example interface, however we can notice that the image query prototype definition is equivalent, even if more user friendly, to specifying an `<object_clause>` whose `<object_predicate>`'s are in conjunction.

7. An Example

Our approach to image retrieval by content (in particular, with respect to the image analysis process) is explained using an example dealing with business graphics. In Fig.3 a graphical image obtained by the business graphics editor is shown. The ARG representation of this image is given in Fig.4. The node alphabet used to generate the ARG representation, is given in Table 1, the branch alphabet in Table 2.

In the example the following basic objects are defined:

frame, text_block, dot_, line_, block

and the relations:

over_text, left_text, below_text, below_dot, below_block, in_dot, in_line, in_block.

The following objects are recognized:

number_of_dots (in the frame), **number_of_blocks** (in the frame), **poly_line**, **poly_block**, **graph_text**, **graph_line**, **graph_bar**.

The last three objects are the most complex objects which can be recognized in such business graphics images. The object **graph_text** can have the attributes **title**, **y_title**, **x_title**. The objects **graph_line** and **graph_text** can have the attributes **legend**, **y_value**, **x_value**, etc.

The computation of the object interpretations for this example are given in Appendix 1. In the example, the image clusters could be **bar graph**, **line graph**, **pie graph**, **line-bar graph**.

Referring to the example, a possible query could be:

```

RETRIVE IMAGES (line graph/0.9 ) OR
  (line_bar graph/0.7 )
CONTAINING
  (graph_text/1.0 WITH y_title MATCH "rainfall") AND
  ((AT LEAST 1 graph_bar/0.8 WITH
    (y_value ≥ 70 AND legend MATCH "Italy")) OR
  (AT LEAST 1 graph_line/0.8 WITH
    (y_value ≥ 70 AND legend MATCH "Italy"))
  )

```

With this query the user is looking for graphical image, of type line graph or line_bar graph, containing information about rainfall, and with at least one rainfall measure for Italy higher than 70.

8. Conclusions and Future work

A small demo-prototype has been implemented on a IBM PC/AT computer dealing with business graphics image, generated by a commercial business graphics editor (i.e. the Graph Assistant). The image analysis process has been implemented using Turbo-Prolog, under MS/DOS. The previous example has been generated using this small demo-prototype.

A more powerful prototype is under development. It will run on a SUN/3 workstation, under Unix 4.2. It will be written in C and Quintus Prolog and will use the SUNCORE graphic package.

From the experiences gained from this prototype, we will then add to the MULTOS multi-media document retrieval capabilities a new function for the retrieval by content on the images contained in the multi-media documents. This will imply a revision of the document query language, the modification of the document access structures and the extension of the query processor.

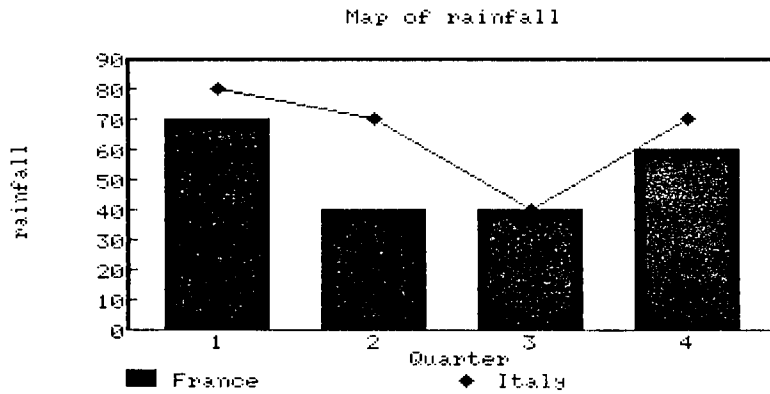


Figure 3: *Example Image*

element	attributes
frame	x1, xd, xv1, xvs, ys
text_block	number text
dot_	number type
line_	number number.1dot number.2dot
block_	number type

Table 1: *Node Attribute Alphabet*

relation	attributes
over_text	number
left_text	number
below_text	number x_dis y_dis
below_dot	number x_dis y_dis
below_block	number x_dis y_dis
in_line	number
in_block	number x_dis y_dis

Table 2: *Branch Attribute Alphabet*

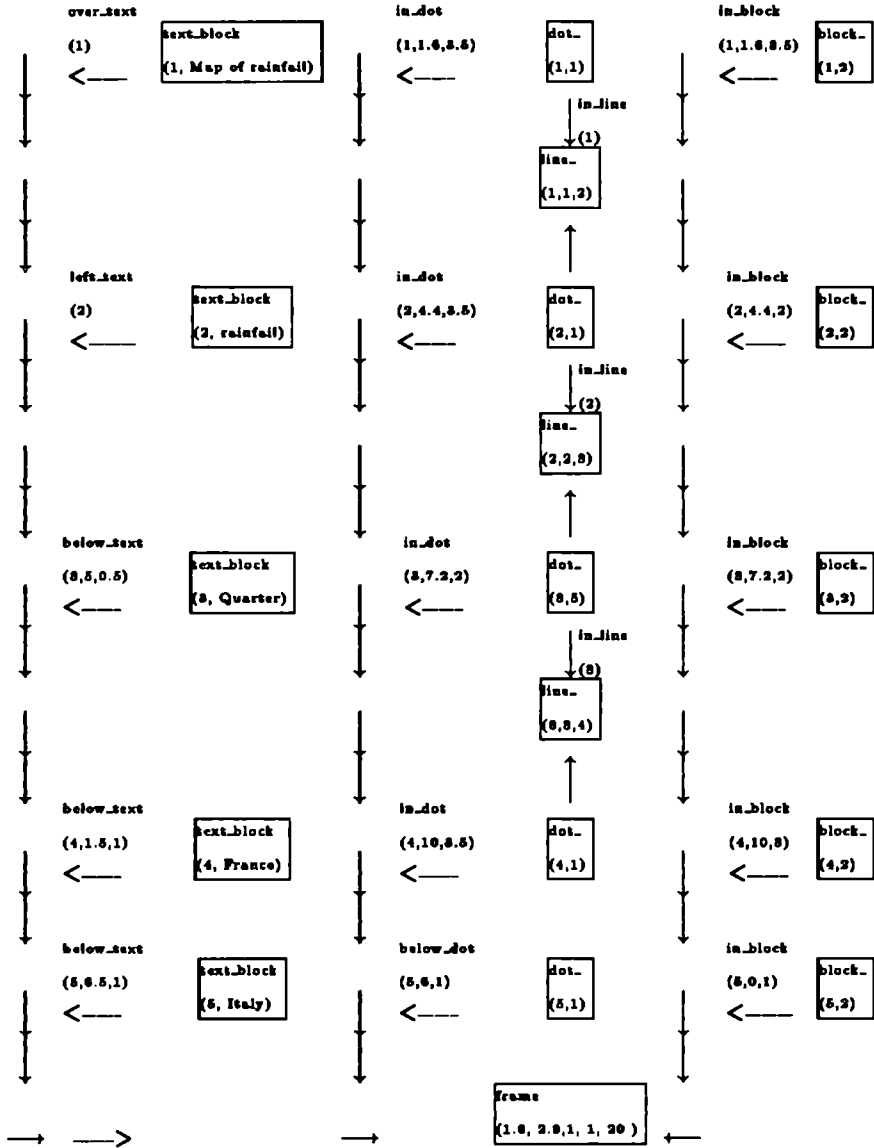


Fig. 4. ARG representation of the example image

REFERENCES

[ANSI84] ANSI X3H3/83-25r3, *Graphical Kernel System*, Special issue, Computer graphics (Febr. 1984)

[Barn81] Barnett J., *Computational Methods for a Mathematical Theory of Evidence*, in Proceedings of the 7-th International Joint Conference on Artificial Intelligence, Vancouver, B.C., pp.868-875 (1981)

[Bert85] Bertino, E., Gibbs, S., Rabitti, F., Thanos, C., Tsichritzis, D., *Architecture of a Multimedia Document Server*, Proc. 2nd ESPRIT Technical Week, Brussels (Sept. 1985)

[Chan81] Chang, S.K., Kunii, T.L., *Pictorial Data-Base Systems*, IEEE Computer Vol.14(11), pp.13-21 (1981)

[Cons86] Constantopoulos P., Yergaroudakis Y., Theodoridou M., Konstantas D., Kreplin K., Eirund H., Fitas A., Savino P., Converti A., Martino L., Rabitti F., Bertino E., Thanos C., Beestra T., *Office Document Retrieval in MULTOS*, Proc. 3rd ESPRIT Technical Week, Brussels (Sept. 1986)

[Econ83] Economopoulos, P. and Lochovsky, F., *A system for managing image data*, Proc. IFIP Congress (1983)

[Eshe84] M.A.Eshera, Fu K., *A Graph Distance Measure for Image Analysis*, IEEE Trans. Syst. Man. Cyber., vol. smc-14, n.3 (May/June 1984)

[Eshe86] M.A.Eshera, Fu K., *An Image Understanding Systems Using Attributed Symbolic Representation and Inexact Graph-Matching*, IEEE Trans. Pattern Anal. Match. Intell., vol. pami-8, n.5 (Sept. 1986)

[Flee84] Fleet, D.J., Jepson, A.D., Hallet, P.E., *A spatio-temporal model for early visual*

processing, Tech. Rep. CSRI, Univ. of Toronto (Canada), 1984

[Gord84] Gordon J., Shortliffe E., *The Dempster-Shafer Theory of Evidence in Rule-Based Expert Systems*. The Mycin Experiments of the Stanford Heuristic Programming Project, Buchanan B., Shortliffe E., (eds.), Addison - Wesley Publishing Company, pp.272-292 (1984)

[Hans78] Hanson, A.R., Riseman, E.M., *Computer vision systems*, Academic Press, 1978.

[Hero80] Herot, C.F., *Spatial management of data*, ACM Trans. Database Syst. Vol. 5(4), pp.493-513 (Dec. 1980)

[Rabi87] Rabitti F., Stanchev P., *An Approach to Image Retrieval from Large Image Databases*, Proc. ACM-SIGIR Conf., New Orleans (1987)

[Rose82] A.Rosenfeld, A.C.Kak *Digital Picture Processing* Academic Press, New York, 1982.

[Salt83] Salton, G. and McGill, M.J., *Introduction to Modern Information Retrieval*, McGraw-Hill (1983)

[Tsic82] Tsichritzis, D. and Lochovsky, F., *Data Models*, Prentice-Hall, Englewood Cliffs, N.J. (1982)

[Tsic83] Tsichritzis, D., Christodoulakis, S., Economopoulos, P., Faloutsos, C., Lee, A., Lee, D., Vandenbroek, J., and Woo, C., *A multimedia office filing system*, Proc. Ninth Int. Conf. on Very Large Data Bases (1983)

[Ullm82] Ullman, J., *Principles of Database Systems*, Computer Science Press (1982)

[Zloo77] Zloof, M.M., *Query-by-Example: A database language*, IBM Systems J. Vol. 16(4), pp.324-343 (1977).

APPENDIX 1.

Result1 : graph bar type A – recognition degree 0.9

Result2 : graph bar type A – recognition degree 0.8

Result3 : graph bar type A – recognition degree 0.7

Result4 : graph bar type B – recognition degree 0.6

Result5 : graph bar type C – recognition degree 0.9

Result6 : graph bar type C – recognition degree 0.5

graph bar type A – recognition degree $\mu_1 = 1 - (1 - 0.9) * (1 - 0.8) * (1 - 0.7) = 0.994$

graph bar type B – recognition degree $\mu_2 = 0.6$

graph bar type C – recognition degree $\mu_3 = 1 - (1 - 0.9) * (1 - 0.5) = 0.95$

Evi1(graph bar type A) = $p_1 = 0.994$

Evi2(graph bar type B) = $p_2 = 0.6$

Evi3(graph bar type C) = $p_3 = 0.95$

$$\begin{array}{lll} c_1 = 0 & c_2 = 0 & c_3 = 0 \\ r_1 = 0.006 & r_2 = 0.4 & r_3 = 0.05 \\ d_1 = 0.006 & d_2 = 0.4 & d_3 = 0.05 \end{array}$$

$$\begin{aligned} K^{-1} &= d_1 * d_2 * d_3 (1 + p_1/d_1 + p_2/d_2 + p_3/d_3) \\ &= 0.00012 * (1 + 165.66667 + 1.5 + 19) = 0.02246 \end{aligned}$$

$$K = 44.5236$$

$$\begin{aligned} Bel(\text{graph bar type A}) &= K * p_1 * d_2 * d_3 \\ &= 44.5236 * 0.994 * 0.4 * 0.05 = 0.88513 \end{aligned}$$

$$\begin{aligned} Bel(\text{graph bar type } A/c) &= K * d_1 * d_2 * d_3 * (p_2/d_2 + p_3/d_3) \\ &= 44.5236 * 0.00012 * (1.5 + 19) = 0.10953 \end{aligned}$$

$$\begin{aligned} Bel(\text{graph bar type } B) &= K * p_2 * d_1 * d_3 \\ &= 44.5236 * 0.6 * 0.006 * 0.05 = 0.00801 \end{aligned}$$

$$\begin{aligned} Bel(\text{graph bar type } B/c) &= K * d_1 * d_2 * d_3 * (p_1/d_1 + p_3/d_3) \\ &= 44.5236 * 0.00012 * (165.66667 + 19) = 0.98664 \end{aligned}$$

$$\begin{aligned} Bel(\text{graph bar type } C) &= K * p_3 * d_1 * d_2 \\ &= 44.5236 * 0.95 * 0.006 * 0.4 = 0.10151 \end{aligned}$$

$$\begin{aligned} Bel(\text{graph bar type } C/c) &= K * d_1 * d_2 * d_3 * (p_1/d_1 + p_2/d_2) \\ &= 44.5236 * 0.00012 * (165.66667 + 1.5) = 0.89314 \end{aligned}$$

The final belief intervals are:

graph bar type A : [0.885 0.890]

graph bar type B : [0.008 0.013]

graph bar type C : [0.101 0.107]

Office Procedures : a Formalism and Support Environment

Sean Baker †, Brian Caulfield †, Remko Hoekstra §, Mauricio Lopez §, Michael R. Rathgeb ¥, Mark Sheppard †, Guy Standen *, Till W. Truckenmüller ¥

The concept of an Office Procedure is presented together with a flexible computerised environment to support the execution of these procedures.

From a detailed analysis of offices in the real world, a set of requirements that need incorporating in the computerised representation of an office task has been derived. These requirements have been used in the design of an office procedure formalism to represent office tasks, and in the design of a computerised environment supporting the execution of office procedures.

A three level model of office procedures and their support environment has been adopted. The model provides easy progression to higher levels for both implementers and users.

The model defines differing complexities and locations of control over the execution of office procedures at each level. Higher levels in the model have been used to define the distributed execution of office procedures, and the total automation of some office tasks.

An aim of our research has been to integrate the representation of office procedures with that of all other information stored in an office, and has led to office procedures being manipulated similarly to other office information server entities e.g. documents. Further, office procedures process and manipulate information stored in an office information server and also use this information for their own control.

Our approach has concentrated on providing simple support, which may be used as the basis of a more powerful model.

1. Introduction

The work reported here is part of the ESPRIT project No. 231 *Design and Operational Evaluation of Office Information Servers* (DOEOIS). The overall objectives of this project are the design, implementation, and evaluation of a family of working prototype office information servers sharing a common external functional interface. Each server provides data management services at two conceptual levels. At the first level data structures are described and the corresponding instances are manipulated using data operations such as *Insert*, *Select* and *Delete*. At the second level, office information processing is supported by *Activities* related to data entities. These activities can, in turn, be grouped into *Office Procedures*. This paper describes office procedure support requirements and a particular approach to office support, addressing some of these requirements.

An *Office Information System* (OIS) should provide tools to support the **analysis and the description** of office work. The analysis can be based on methodologies such as *Office Analysis Methodology* OAM [SUTH 83, SIRB 83] or *Critical Success Factors* CSF [BULL 81].

† Distributed Systems Group, Dept. of Computer Science, Trinity College, Dublin 2, Ireland.

§ Centre de Recherche Bull, c/o IMAG, BP 68 ,F-38402 Saint-Martin D'Hères Cedex, France.

¥ Universität Stuttgart, Fraunhofer Institut für Arbeitswirtschaft und Organisation (IAO).

* International Computers Limited, Lovelace Road, Bracknell, Berkshire, England.

The description can be done using models like *Information Control Nets* ICN [ELLI 79] or *Predicate Transition Networks* PrTN [RICH 84]. The same model can be shared by the analysis methodology and the actual information management system supporting office applications. The OIS should also provide support during the execution of office procedures. This concerns monitoring control and information flow, providing information about the state of procedures or individual activities, evaluating execution preconditions, automatically starting activities under certain conditions, etc. Such a service can be used by managers to gather statistical information about task durations, personnel assignments, etc., so as to be constantly and rapidly informed of the state of work in progress, to simulate the effects of reorganisation, etc. It can also benefit office workers through guidance and integrity support, and relieve them of simple and repetitive tasks.

The office procedure support provided by the server is of a more basic nature than that provided by a complete OIS. This is because the server is one component of a total OIS and its information storage and retrieval services are therefore designed to be used by the OIS to build a more sophisticated environment.

The office procedures model adopted is intended to describe activities and procedures as part of the conceptual schema of a given application. Activities and procedures are viewed as dynamic entities. Therefore, classes of these dynamic entities can be defined whose instances correspond to particular tasks to be executed. Each procedure and each activity is related to an entity of the conceptual schema, called its *Focal Entity*. The focal entity determines the information context of procedures and activities, thus establishing an explicit relationship between the static and the dynamic elements present in the conceptual schema.

The model and corresponding execution support are provided at three levels in order to allow modular and incremental development of applications. This layered approach aims to meet the requirements of office applications where the structure of the procedures is loose or difficult to establish and can evolve through time. At level 1, only activities exist and activity classes are defined independently of each other. An *enabling precondition* is related to each activity class. When this precondition becomes true, the activity becomes enabled and can be executed. At this level activities are bodyless, i.e., the server only provides information about the state of activities but the application is responsible for providing and executing the operations corresponding to their body. At level 2, an additional feature is provided; a computer program is associated with each activity and can be either started automatically by the server or manually by the user. At level 3, activities can be grouped into procedures and precedence relationships established among them. In a similar manner to activities, an enabling precondition can be related to each procedure.

Individual activities can be given the properties of atomicity, isolation and permanence of effect through the use of transactions. However, these properties can not be given to an entire procedure or sequence of activities, so as to avoid the problems of holding locks for long periods of time, or undoing large amounts of work. The locking mechanism is also used to guarantee that critical terms in the enabling precondition of a procedure/activity are still true at the moment of effectively starting execution.

The remaining sections of this paper describe 1) some important requirements of an office procedure formalism and support environment; 2) those requirements addressed within the project, and which thus form the rationale for the approach taken; 3) some technical features of the three level model adopted, and; 4) conclusions regarding the work and some possible extensions to the model to form the basis of a complete OIS.

2. Requirements Analysis

Current office system support tools are designed to improve the performance of individual tasks common to most offices [CROF 84]. They are designed on the basis of evolved technological capabilities and address specific support problems. However, such traditional "automation through technology" approaches are of limited applicability because of the relatively unstructured and people-centered behaviour in offices [BAIR 86, CONR 85]. Significant improvements of efficiency and effectiveness can only be achieved, when the design of office support tools is based on an understanding of all the important characteristics of offices and of office work [BRAC 84, BRAC 86, TRUC 86a, TRUC 86b].

Offices are, above all, "open systems" due to the demand for information exchange with their external environment [HEWI 86], their high degree of inherent parallelism, their incremental evolution and their distributed nature [HEWI 82]. Other characteristics that need to be recognised are the organizational complexity, the importance of temporal aspects, the existence of office procedures, application levels, usage characteristics, etc.

The design of office support systems should be based on analysis of how tasks are performed, as well as why they are performed [HIRS 86, NIEM 87]. Furthermore, future developments need to be considered, as well as current practices, to ensure that the needs reflected in the support facilities will remain valid when the system becomes generally available. In an analytical perspective which sees the office as an environment for the performance of largely formal and structured functions, the interpretivist perspective also needs to be taken into account. This conceives the office in terms of predominantly unstructured and informal human interactions (social issues) [HIRS 86]. An analysis of the socio-technical issues [HIRS 86, NIEM 87, TRUC 86a, BRAC 86] has formed the basis of the guidelines for designing office support systems [RATH 87].

An important part of the office support problem concerns the requirement for a flexible support environment for information handling within office procedure structures. In the context of office procedures, characteristics that should be taken into account include decision-making processes, the performance of a possibly long lived series of activities within well, or poorly defined execution schemata, parallelism, distribution, the possibility of unexpected obstacles and the shared performance of office procedures by several interacting people [REIM 86, TRUC 86a].

Support for office systems needs to address the following major requirements [ZANG 86]:

- the **correct information** should be provided;
- at the **correct time** and;
- at the **correct location**.

Provision of **correct information** addresses the filing and retrieval problem. The growing volume of electronically accessible information forces a need for efficient facilities for the storage of this information. This needs to be done in a manner that matches the access structures and behaviour conditioned by particular office functions and staff. Likewise, retrieval needs to be supported, not only allowing direct retrieval (via keywords, thesauri, etc.), but also indirect retrieval in the context of office procedure execution. Such support facilities are based on the concept of a focal object [SIRB 83, SUTH 83], which relates all information critical to a given procedure or activity instance, and establishes the necessary link between the first and second conceptual support levels mentioned above. Support facilities should also exist for the evaluation and notification of information related task execution conditions, querying the current state of active procedures, and so on.

The concept of **correct time** is directly related to the nature of office activity as a chronological sequence of events. It addresses such familiar notions as versioning, history management, and the treatment of time and date information. Time related conditions are an essential component of most office procedures.

Delivery of information at the **correct location** addresses the need to take the distributed aspects of office procedure support into account. Complex office tasks need to be subdivided into a number of logically complete sub-tasks that can be assigned to individual office workers, possibly at different locations. Adequate support is therefore required for cooperation between individual procedures and associated office workers/work places, to guarantee that given tasks will be completed effectively and in the right order. Such support needs to include the means of integrating individually defined and separately performed task units into an overall procedure. An important requirement that needs to be addressed by office procedure support must, therefore, be the routing of information to the proper location. Widely unstructured (problem solving) tasks may be solved within a very loosely defined activity framework (e.g. time scheduling and reminder functions) where only a few activities can be performed without human interaction and decision. Well defined (repetitive) tasks may be extensively, or even totally, performed within an execution schema that predicts the structure of activity performance to a high degree of precision. However, both types of office work, unstructured problem solving tasks, and well defined, repetitive tasks, require the routing of information to the correct location.

A good approach to office systems support first addresses problems where great improvement can be achieved with little effort, and then stepwise augments the solution for more complex problem areas. Office work is performed within an evolving dynamic environment, which creates the necessity not only to provide support for specific situations within the range of office work, but also to provide functionalities and concepts that support the office worker in the redefinition of task structures. Such support ought to allow the restructuring of office tasks as experience concerning problem solution schemata grows.

The requirement then, is to support both loosely and well structured tasks. However, good support should also permit the incremental migration of task knowledge from user to system as experience in a given problem domain increases. At the base of this continuum, a primitive but flexible set of task support functions is required, while at the top, a higher level, more abstract language is required.

3. Support for Office Procedures

3.1 Rationale for the Support

The rationale behind the design of the office procedure formalism and support environment is based on the key characteristics of simplicity, flexibility, integration to data models, and a logical application development path.

Simplicity is the prime objective, whilst retaining enough functionality to represent a diverse spectrum of offices. A simple model facilitates easier multi-vendor implementation of the support environment and the provision of easily learnt and understandable systems. This approach facilitates the construction of more complex models (as and when they are developed) from the simple initial model, thereby maintaining backward compatibility through the same underlying model.

The flexibility of the model is important for a number of reasons. Not only are the tasks in a single office very diverse, but the offices themselves may be very different. The office procedure formalism needs to represent both structured and unstructured tasks equally well. The three level model has been found sufficient to address this problem. With little or no practical implementation of office procedures to-date, it is important to produce a flexible model which will not have to be extensively altered as implementation experience accumulates.

Level one of the model delegates much of the control and structure of the office procedure to the application domain i.e. the client. This allows the client to independently specify arbitrarily complex control strategies and to structure the activities making up the procedure with little hindrance from the model. At level three much of the control and structure of the office procedures is incorporated into the formalism and support environment allowing sufficiently well structured tasks to be completely specified and executed.

Office procedure consistency, integrity and portability are seen as important aspects of the office procedure formalism. The ability to use the formalism to, unambiguously, specify an office task has been recognised as an important aspect of any office procedure model [BRAC 84]. These needs have resulted in a high degree of integration between the formalism and the data model used to represent the information in the office. The design of the formalism allows office procedures to be modelled using the data model and the instances of procedures to be represented in the associated database. The aim has been to integrate the data model with the office procedure formalism in a manner such that all information used and manipulated by the procedures is maintained in the database, whether for triggering and controlling office procedures or that processed by the procedures themselves.

The consistency and standardisation afforded by the use of a formal data model will ensure easy interchange of office procedures and the information used by those procedures both inter and intra office, in the same manner as the more static components of the office application schema.

To be effective, office procedures need to be able to handle not only the ubiquitous record but also the office document [BRAC 84]. The integration with a data model capable of representing most

office information types including office documents addresses this need. The data model selected for the project is the Fact Model [SACC 86], it is an irreducible, semantic data model which fulfils this requirement.

Ballou & Kim [BALL 84] suggest that office automation systems require a high level of "user friendliness" as the office automation system operator and end user are normally the same person. In a way analogous to views it is possible to provide office procedure interfaces tailored to individual users' requirements (natural language, graphical etc.) supported by the single consistent conceptual schema of the underlying data model. Support for local naming conventions etc., can likewise help to provide a more familiar environment for the user whilst maintaining consistency.

A further incentive to integrate the office procedure formalism and the support environment with the data model and database is that many of the functions familiar to both (conditions, transactions, locks) need not be duplicated in the final office information system. For example precondition evaluation can be partly implemented by qualified queries over the database triggered by updates, thus reducing implementation effort.

The multi-partner nature of the project and eventual multi-vendor implementation of an office information system, makes implementation flexibility a key issue. The model is designed with this in mind, giving implementers a wide choice of hardware and operating regimes. The project is currently being implemented on three different hardware bases and using two different operating regimes (UNIX & VME). The three level model is designed to allow differing but compatible levels of implementation effort and provide an implementation upgrade path to higher levels. This is achieved by making the lower levels of the model compatible with the higher levels and thus implementing the higher levels just builds on the lower levels. In fact it is possible to simulate the higher levels using the lower levels.

A feature of the model is that office procedures developed at the lower levels are compatible with the higher level models and can be used unmodified at the higher levels. The reasoning behind this facility is to provide a modular approach to producing office procedures. Thus an office procedure can be constructed from a library of lower level activities (and possibly sub-procedures) plus the required overall control. At the very lowest level of the model, an activity is considered to be a black box with an interface to the office information system via shared information in the database. In this way, it is hoped that existing office automation products (eg. Payroll, Accounts etc.) may be incorporated into an office procedure as activities with little or no updating.

To aid the development of office procedures, the three level model allows a user to transfer the control and structure developed directly at level 1, to the primitives supplied at the higher levels. Thus clients may develop informal office procedures at level one and later after testing and trials formalise them within the higher levels of the model.

3.2 The Basic Concepts

The basis of the support centres around two concepts, the activity, and the (office) procedure. Office procedures are considered to be relatively long-lived tasks, involving a series of steps, or sub-tasks, which collectively achieve some definable goal. These steps are small "sensible" units of office work, and their computer supported equivalents are referred to as "activities". What constitutes "sensible" in the context of this definition is the responsibility of the application designer, but most importantly, activities should be sufficiently small so that they could be restarted from the beginning without any great loss.

It is certainly the case that many office tasks are repeated many times, with different items of data. For this reason, all levels of the model contain the concept of classes and instances of procedures/activities. An activity class is a definition of a sub-task; it specifies the type of processing performed, the conditions under which it should be performed, and the type of data manipulated. It is a "blueprint" for creating instances of the activity class. The activity instances perform the actual information processing. Each activity instance operates on a different set of data. So, an activity class may have many instances, but each instance may be associated with only one class. The same applies to office procedures, the classes are used to generate instances which perform the actual data processing.

Associated with each procedure and activity class, and fundamental to all levels of the model, are

- a precondition, and
- a focal entity type.

The precondition specifies the conditions under which it is appropriate to create and execute an instance of the procedure/activity class. The precondition consists of a number of individual conditions, called terms, which may reference any entity in the underlying database. In particular, the terms may refer to

- (i) the state of any entity in the database, or
- (ii) operations on any entity in the database.

Examples of the two types of term, in the syntax which has been adopted are:

- (i) **test** timecard **where** timecard.status = "filled_in"
- (ii) **test** insert user **where** user.space_allocation > 200

The meaning of the first term is that an instance of the procedure/activity may be created and run whenever a timecard with the status "filled_in" is found. In the second case, an instance will be created whenever a new entity of type user, with a space allocation greater than 200, is inserted in the database. A number of both kinds of term may be **anded** and/or **ored** together to form a more complex precondition. Requirements analysis in real-world offices suggests that the complexity of preconditions will usually be two or three terms at most. The syntax adopted for preconditions closely matches that of the Fact Model Query Language, enhancing the integration of the office procedure support with the data model.

The focal entity type specifies the type of data manipulated by the class of activities/procedures. It is similar to the concept of focal object proposed in OAM [SIRB 83, SUTH 83]. Each activity instance has a focal entity which is an instance of the focal entity type for that class. The focal entity is:

- (i) a Fact Model entity,
- (ii) related to all data critical to the activity/procedure via FM facts, and
- (iii) a parameter to the activity/procedure instance.

The focal entity must be unique to one instance of a particular class of activity/procedure. This means that the focal entity may be used to uniquely identify a particular instance of an activity or procedure class.

The precondition and the focal entity type are closely related to each other. Essentially, evaluation of the precondition involves searching the database for instances of the focal entity type for which the precondition holds. For each such instance, an instance of the related activity or procedure class will be instantiated.

3.3 The Three Level Model

The three levels of the model provide a hierarchy of support for both users and implementers. Progression through the three levels for the user involves placing an increasing amount of both control and effort within the system. The reduced amount of control for the user means that flexibility for handling poorly understood tasks is reduced, but the level of support for well defined, structured tasks is significantly increased. Implementation of succeeding levels of the model involves enhancement of the lower levels, though each level could be implemented independently.

In outline, at level 1 there is no concept of an office procedure in the system, only independent activities. The activities have no predefined action associated with them. In essence level 1 provides a powerful system to evaluate preconditions and identify focal entities for which the precondition is true. Users have complete control of what action is taken on satisfaction of the precondition and when it is taken. Level 2 enhances this support with a predefined action known as a "body" which can be executed either automatically by the system when the precondition is

satisfied, or at the discretion of the user. There is still no notion of procedure, though informal office procedures could be written using activities whose preconditions are logically connected. Level 3 introduces the concept of office procedures which are composed of a network of level 1 and 2 type activities. Permanence of control is provided so that procedures can survive crashes to the granularity of activities, i.e. activities running when a system failure occurs must be restarted. A language, based on the control structures of high level languages is proposed for the definition of office procedures. The following three sections provide a somewhat more detailed, though far from complete, description of the three levels.

3.3.1 Level 1

Level 1 provides a minimum level of support for office tasks. It is flexible, but rudimentary. Since the system has no notion of office procedures at this level, none of the linkages between activities need to be automated. At this level the definition of an activity class consists of the following parts:

- **a class name**
this identifies the class and must be unique to it. It is also used in conjunction with the focal entity to uniquely identify any activity instance.
- **a precondition**
this is exactly as described in section 3.2
- **a focal entity type**
as described in section 3.2. This is a type in the Fact Model specifying the "special data item" for the class. It should aggregate (i.e. have facts linking it to) all the critical data of the activity class. Sometimes it might be necessary to artificially add such a type to the database schema.
- **a description**
an informal text description of the activity class.

The definitions of various activity classes are stored in the office information server as FM entities, and are used to create instances of the class as previously described. The instances themselves are also stored in the server, and in addition to their association with a particular class each instance has associated with it

- **a focal entity:**
an entity of the focal entity type unique to this instance within the class; with the class name it uniquely identifies any instance of any class;
- **a status:**
the status may be either uninstantiated, enabled, started or finished; transitions among these states are caused by the truth value of the precondition and by the primitives provided for manipulating activities.

When the precondition for a class is found to be true, for an instance of the focal entity type, an instance of the class is created and its status changes from uninstantiated to enabled. The enabled instance is then placed on a list of all enabled activity instances. A user may obtain an enabled activity instance by performing a **start** call:

```
start activity where activity.activity_class.name = <activity class name>
and activity.focal_entity = <expr>
```

As a syntactic convenience, we allow the term "focal_entity" to be replaced by the actual FM entity type, with an asterisk to identify this, e.g., "person*", if person is the focal entity type for a particular activity class. The syntax for all the primitives closely matches that of the Fact Model Query Language; however, the **start** primitive operates on an instance-at-a-time basis, whereas other FMQL statements are set-at-a-time. This provides the office worker with a flexible choice of tasks, since s/he may choose a specific activity, or any of a specified class, or one of any class for a given focal entity, by appropriately specifying the **with** clause of the query. When the **start** is performed, the status of the activity instance returned changes to started and the caller may then take whatever action is appropriate.

On completion of the activity, the caller issues an **end** activity statement, thus informing the system of the completion of the task. Instances may also be terminated abnormally using the **abort** primitive:

abort activity **where**.....

Again, by appropriately specifying the **where** clause, the caller chooses which instances to abort. Note however that the **abort** operates on a set at a time basis, since it is useful to allow privileged users to abort, say, an entire class. Aborted instances are returned to the enabled list if their precondition still holds and to uninstatiated otherwise. The transition diagram of figure 3.1 shows the movement of an instance among states.

Two further primitives, **load** and **unload** are provided for the purposes of initiating and terminating the evaluation of a classes precondition. Once a class definition has been added to the system (using normal FM DML statements), for example:

```

insert activity_class with class_name = "WashCar"
                        and focal_entity = car
                        and precondition = "test car where car.status = dirty"
    
```

loading causes the evaluation of the precondition to commence. Unloading performs the reverse function without deleting the class from the system.

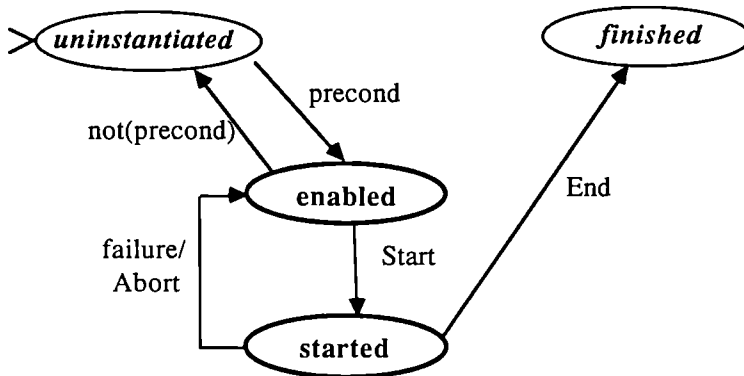


Figure 3.1

3.3.2 Level 2

In level 2, support for better understood tasks is improved by the association of a predefined action with an activity class. The form of the action itself is application dependant and it is not intended that an application language will be developed. For this reason, the "body" of the activity is considered to be a file containing executable code which may be run once the precondition has been met for an instance of the focal entity type. The identifier for the focal entity is passed as a parameter to the program, and the code is executed once for each focal entity for which the precondition is true.

With the addition of a predefined action it is also necessary to specify under what conditions it should be started. Two modes of execution are supported, manual and automatic. For an activity class for which the manual mode is specified, a list of enabled instances is maintained just as is done at level 1. Execution of the body is then initiated at the request of the user through the use of the **start** primitive. For automatic activites, the action is executed immediately on satisfaction of

the precondition with no user intervention whatsoever. This allows the complete automation of some tasks while permitting the user more control over others, such as interactive ones.

In a distributed environment a further question with regard to the action is where the code should be executed. For this reason, the specification of the body breaks down into two parts

- a filename, and
- a sitename.

The filename specifies the name of the file containing the program to be executed, the sitename specifies the execution location. In the case of manual activities, the program must always execute at the site of the caller, thus the sitename must be left null. For automatic activities, the application designer may choose a site at definition time for the activity class, and this is specified by the sitename. Clearly it is the responsibility of the applications designer to ensure that the action is executable at all potential sites. The use of this method permits a degree of flexibility so that office workers need not be constrained to perform set tasks at times and locations dictated by the system, and increased flexibility in this area is envisaged as the system develops.

3.3.3 Level 3

Level 3 of the model introduces the idea of procedure to the system. As described in section 3.3.1, informal office procedures can be written at the lower levels, but the system itself sees only individual autonomous activities. Level 3 automates the connections between a network of Level 1 and 2 type activities so as to provide permanence of control over office procedures. At the prototype stage, the support will concentrate on flow of control and information for well structured tasks, while the provision of good exception handling facilities is deferred until a later phase of the project. The adaptation of running office procedures is not considered at this point either.

As previously described, an office procedure consists of a network of Level 1 or 2 type activities connected by precedence information. The task of Level 3 is largely the specification of this precedence information, the activities which make up the procedure, and information which associates the focal entities of the various activities with that of the procedure itself. For example given the activity of the example above, "WashCar", with focal entity "car", in a particular procedure the activity might be required to process only the "car" belonging to the "person" who is the focal entity of the procedure. This can be indicated as follows

WashCar (car : person*.car)

where person* is the focal entity of the procedure.

All of the above information is contained in the execution schema for the procedure. The execution schema is written in a language, known as OPL3, specially designed for the purpose. The language must provide control structures to allow for all of the following

(i) sequencing

it must be possible to write a sequence of activities to be run in succession. This is done by writing the activities separated by semi-colons

<activity₁> ; <activity₂> ; . . .

and is known as a statement list. Other statements of OPL3 may also form part of a statement list.

(ii) parallelism

a number of statement lists may be run in parallel using the **cobegin** statement:

```

cobegin
    <statement list1>
    || <statement list2>
    || ...
coend

```

The execution of the statement following the **coend** commences only when all of the statement lists in the construct have been completed.

(iii) choice

a choice between a number of possible paths can be made using either of two statements:

```

await
    <statement list1>
    || <statement list2>
    || ...
awend

```

This statement non-deterministically chooses one and only one of the statement lists. In effect the statement waits until some statement list is executable.

```

switch
    <cond1> : <statement list1>
    || <cond2> : <statement list2>
    || ...
swend

```

The **switch** statement chooses zero or one of the statement lists based on the associated conditions which are basically similar to those of a precondition. The conditions are evaluated in order of appearance, so if more than one condition is true, the first encountered is executed. The **switch** does not wait, the conditions are evaluated once only.

(iv) Looping

while it is possible to construct loops in the system without an explicit looping statement, the provision of one should considerably aid the task of writing complex schemas. The syntax is

```

while <cond> do <statement list> whend

```

and the statement list is executed so long as the condition is true, just as for a HLL while statement.

Through the use of these control structures, it is possible to write any possible schema required. The statements of the language have a distinct programming flavour to them which is in keeping both with the orientation of the support towards application designers, and the often sequential nature of individual office tasks, whereas, office work in general exhibits a high degree of parallelism.

For the user, primitives similar to those for level 1 and 2 activities are provided to permit the starting, aborting, etc., of procedures, and again the syntax of these primitives matches that of the normal FMQL statements.

4. Conclusion

The three level model approach to supporting office tasks provides a simple, flexible and extensible environment in which a range of structured and unstructured office tasks can be modelled. Further it provides a development path along which some unstructured tasks can be formalised as their structure evolves. These modelled tasks (office procedures) are represented within a data model

framework which provides easy, consistent and integral access to information which they manipulate, and allows easy management, inspection and manipulation of these procedures themselves.

Work is now proceeding to implement the proposed support, and it is hoped that a rough prototype will be demonstratable by October of 1987.

Extensions to the model yet to be researched include the integration of; actors/roles, the distributed aspects of office work, sophisticated exception handling, the user/client interface, and the integration with a full office information system.

ESPRIT project 834 (COMANDOS) is researching the low level support of office systems and as such is researching the implementation of an architecture which could support the model described, providing a second testbed.

References

- BAIR 86: Bair, J. H.: Determining Business Value for Advanced Office Systems; in: AFIPS, Office Automation Conference, March 1986, Houston, Texas, pp. 283 - 289
- BALL 84: Ballou, D., Kim, S. : A Systems Lifecycle for Office Automation Projects; Information and Management, 7.
- BARB 83: Barber, G.: Supporting Organizational Problem Solving with a Work Station; in: ACM Transactions on Office Information Systems, Vol. 1, No. 1, Jan. 1983, pp. 45 - 67
- BRAC 84: Bracchi, G., Pernicci, B.: The Design Requirements of Office Systems; in: ACM Transactions on Office Information Systems, 2, No. 2, April.
- BRAC 86: Bracchi, G.; Pernici, B.: Trends in Office Modeling; in: Office Systems, A. A. Verrijn-Stuart, R. A. Hirschheim (eds.), Elsevier, IFIP, 1986; pp. 77 - 97
- BULL 81: Bullen, C. V.; Rockart, J. F.: A Primer on Critical Success Factors; Centre for Information Systems Research, Sloan School of Management, Massachusetts Institute of Technology, 1981
- CONR 85: Conrath, D. W.: Office Automation: The Organization and Integration; in: AFIPS, Office Automation Conference, Feb. 1985, Atlanta, Georgia, pp. 303 - 312
- CROF 84: Croft, Bruce, W. and Lefkowitz, Lawrence, S.: Task Support in an Office System; in: ACM Transactions on Office Information Systems, Vol. 2, No. 3, July 1984, pp. 197 - 212
- ELLI 79: Ellis, C. A.: Information Control Nets: a Mathematical Model of Office Information Flow; in: ACM Proc. Conf. Simulation, Modelling and Measurement of Computer Systems, Aug. 1979, pp. 225-240
- HEWI 82: Hewitt, C.; de Jong, P. (1982): Open Systems; Massachusetts Institute of Technology, AIM 691, Dec. 1982
- HEWI 86: Hewitt, C.: Offices are Open Systems; in: ACM Transactions on Office Information Systems, Vol. 4, No. 3, July 1986, pp. 271 - 287
- HIRS 86a: Hirschheim, R. A.: Understanding the Office: A Social-Analytic Perspective; in: ACM-Transactions on Office Information Systems, Vol. 4, No. 4, Oct. 1986, pp. 331 - 344

- HIRS 86b: Hirschheim, R. A.: Perspectives and Views of the Office: Alternative Approaches to Understanding the Office; in: Office Systems, A. A. Verrijn-Stuart, R. A. Hirschheim (eds.), Elsevier, IFIP, 1986; pp. 29 - 56
- NIEM 87: Niemeier, J.: Topical Trends as to the Methods of Planning and Designing Information and Communication Systems (in German); in: ONLINE '87, 10 th European Congress Fair for Technical Communication, Kongress VII, Feb. 1987, pp. 31.1.01 - 31.1.22
- RATH 87: Rathgeb, Michael, R.; Friedrich, Rainer; Truckenmüller, Till, W.: Guidelines for the Office Information Server (OIS) Model and Future Office Information Server Functionality; Deliverable ESPRIT Project 231 (DOEOIS), March 1987
- REIM 86: Reim, Friedemann: On the Role of an Office Information Server; Working Paper IAO-FRR-8606-1, ESPRIT Project 231 (DOEOIS), June 1986
- RICH 84: Richter, G.: Netzmodelle für die Bürokommunikation; Teil 2; in: Informatik Spektrum 7, 1984, pp. 28-40
- SACC 86: Sacco, G. M.: The Fact Model: A Semantic Data Model for Complex Databases; ETW Conf. Proc. 1986, also: ESPRIT Project 231 (1B2 Report), Feb. 1986
- SIRB 83: Sirbu, M. A. et al.: Office Analysis: Methodology and Case Studies; Massachusetts Institute of Technology, Lab. for Computer Science, MIT/LCS/TR-289, Cambridge, Massachusetts, 1983
- SUTH 83: Sutherland, J.: An Office Analysis and Diagnosis Methodology; Massachusetts Institute of Technology, Lab. for Computer Science, MIT/LCS/TR-290, Cambridge, Massachusetts, 1983
- TRUC 85: Truckenmüller, T. W.; Ness, A. J.; Niemeier, J.: An Analysis Method for Office Automation; Deliverable ESPRIT Project 231, DOEOIS, FhG - IAO, August 1985
- TRUC 86a: Truckenmüller, Till, W., Rathgeb, Michael, R., Niemeier, J.: Analysis on Information Handling and Manipulation Activities in Offices; Deliverable ESPRIT Project 231 (DOEOIS), August 1986
- TRUC 86b: Truckenmüller, T. W.: Methodological Issues for the Design of an Office Information Server - Focal Topics for the Analysis from an Office System Perspective; in: ACM Conference on Research and Development in Information Retrieval, Rabitti F. (ed.), Sept. 1986, Pisa, Italy
- ZANG 86: Zangl, H.: Transparenz der Zusammenhänge im Büro-Durchlauf-zeitenanalyse zur Untersuchung und Gestaltung von Arbeitsabläufen; in: Bullinger, H.J. (ed.); Büroforum '86, Informationsmanagement für die Praxis, Berlin, Heidelberg, New York, Tokio, 1986

Techniques for Filing and Retrieval of Office Information

Gordon McAlpine

Dansk Datamatik Center
Lundtoftevej 1C
DK-2800 LyngbyABSTRACT

Future Office Information Systems (OIS) will be able to store and retrieve nearly all the data that are used in offices. The objective of Project MINSTREL has been to identify and develop software techniques that are necessary to provide convenient filing and retrieval facilities within such an OIS. This paper describes the main achievements of Project MINSTREL, which finishes in September 1987.

The central achievement of the project has been the design of an Office Information Model that provides a uniform data representation for all the application programs in such an OIS, and thus supports data integration. The resulting object-oriented model includes constructs for conveniently representing the varied and complex structures of office data, for representing data in accordance with other data models, for powerful text retrieval operations, for representing imprecise information, for specifying and enforcing constraints and for defining inheritance. The model has been formally specified, and a subset has been implemented. The paper summarises this model, our experiences in using it and its relevance for future OISs.

The other main problem addressed in MINSTREL has been "effective retrieval", i.e. how can the system come as close as possible to retrieving exactly those data objects which are relevant to the user's queries? The paper includes a description of 3 techniques that address this problem. Firstly, we have shown that effectiveness of text retrieval can be improved by employing syntactic processing, and that this approach is worth further investigation. Secondly, we have developed a technique for handling missing and incomplete information - both in the user's query and in the stored data. This technique ranks objects in relation to a measure of the certainty with which they fulfill the query criteria. Thirdly, we have designed a user interface that employs graphics and direct manipulation to meet the usability requirements for an office filing and retrieval facility.

1. Introduction

The objective of Project MINSTREL (Models for INformation, STorage and REtrieval) has been to research and develop software techniques that should be incorporated into future office filing and retrieval systems.

Recent technological developments, particularly in the areas of optical disks and optical character recognisers, have made it technically feasible to store nearly all of the information used in offices in electronic form. The quantity of data storable is immense, and, in addition, these data may vary both in the media used to communicate them, e.g. voice, image and text, and in their form, e.g. documents, records and messages.

The central problem being addressed in MINSTREL is the design of filing and retrieval facilities that provide effective access to such a large and varied data repository.

We have taken the view that offices can be characterised as information processing systems, since most office activities revolve around the creation, retrieval and manipulation of information. Effective access to information is thus a crucial requirement for office support systems.

In MINSTREL, we have investigated access to office information at two levels. At the system level, we have designed an office information model that provides a uniform data representation for all office data. At the user level, we have researched techniques for filing and retrieval facilities to be used by office workers to retrieve data from an office information system.

The construction of an office information system that supports access to all office data poses crucial requirements to the system's data representation. In particular, the requirement for data integration, which most current office information systems fail to meet, becomes even more important. By data integration we mean that data created by each office support tool (i.e. application program) within the system should be available to all the other tools.

This requirement has led us to the conclusion that future integrated office systems that attempt to store all office data should be based on a uniform data representation. A data representation formalism that meets these requirements must be capable of describing the properties of and operations upon all forms of office data. In MINSTREL, we have called this formalism an office information model. The language of our model can be used to describe the information that is processed in the domain of a specific office.

A central research task in MINSTREL has been to develop such a model. The Office Information Model has functioned as a focal point for all the other research areas of the project. The MINSTREL Office Information Model is described in section 2.

To define what we mean by office filing and retrieval facilities, we must first make our assumptions clear.

We have taken the approach that office workers prefer to organise office data as a collection of data objects - distinguishable units of information that have a persistent and independent existence (often referred to as entities in data modelling). By "persistent" we mean that the objects are of long-term value to the user, and should therefore persist from one activity to the next. By "independent" we mean that data objects have a meaning and a lifespan that is independent of the existence of other objects. Examples of data objects are a letter, a form, a spreadsheet.

We assume that an office worker will have a need to know certain things that he does not already know, and that an office information system provides a repository of data objects that may contain information relevant to such information needs. He therefore expresses his information need in the form of a request to the system. Note, however, that a request will typically be an imperfect expression of a user's information need, and that only the user can tell whether an object contains the information he is seeking.

We then define the retrieval facility of an office information system to be the set of tools that a user can utilise to retrieve data objects that are likely to be relevant to his information need as expressed by his request.

To attack the problem of how to provide an effective retrieval facility we have investigated three areas:

- the design of a highly usable user interface to an office filing and retrieval facility
- the handling of imprecision, i.e. missing or incomplete information, both in the user's queries and in the stored data
- the investigation of whether the effectiveness of the retrieval of text within an office retrieval facility can be improved by the use of linguistic processing.

The results of the research in these areas is reported on in section 3 of this paper.

Section 4 reports on a prototype implementation of a subset of the MINSTREL office filing and retrieval facility. Section 5 describes our conclusions from MINSTREL.

2. The Office Information Model

The main requirements for our Office Information Model were:

1. to provide powerful data description facilities for office tools, so that as much of the domain information as possible is represented declaratively in the stored data base and schema
2. to provide for the representation of all forms of office data, including multi-media data, complex object structures and imprecise data, in an integrated way
3. to provide means for representing data so that they conform to data models other than our Office Information Model
4. to provide convenient access to meta data
5. to provide powerful retrieval operations, especially for text.

In a pilot project prior to MINSTREL, we investigated the state of the art in data modelling. We quickly came to the conclusion that the classical data models, such as the relational data model, were clearly not capable of fulfilling these requirements. We then turned our attention to the field of conceptual (or semantic) data modelling, as defined in [Brodie 84]. In particular, we fetched a lot of inspiration from DAPLEX [Shipman 81], which is a so-called functional data model.

Conceptual data models were originally intended as modelling tools to be used by system analysts and designers during the design phase of system development. As in DAPLEX, we have taken the approach of providing an implementation language that directly supports our conceptual data model. This eliminates the need for translating the system specification from a conceptual to a classical data model (which is typically used for implementation), and thus speeds up the system development process. Further, it improves the maintain-

ability of the implemented system, since it is expressed in terms of a higher-level model that captures more of the application's semantics in the declarative part of the system, i.e. the schema, as opposed to the imperative part of the system, i.e. the application programs.

The remainder of this section gives an overview of the features of our Office Information Model and the reasoning behind them. Technical details are not given, due to the restricted length of this paper.

2.1 Overall Approach and Basic Features

The description of data structures and the static properties of data were thoroughly researched at the beginning of the project. An initial model, as described in [Dunnion 85], was then designed and implemented in a prototype. Experience gathered from implementing a prototype filing and retrieval facility using the prototype model was put to use in the specification of a revision of the model.

The initial model was similar to traditional Database Management Systems in that it was accessed from application programs by means of its own data definition and data manipulation languages. In the revised model we decided to integrate our data modelling constructs within a programming language, and thereby provide an (office) database programming language. The advantage of this merge of database and programming language technologies is that programs can manipulate internal, transient data structures and persistent data objects in exactly the same way, the only difference being that the latter are stored permanently.

In the initial model, data were passed back to (Prolog) application programs as rather rigid and crude Prolog data structures. This led to much of the application code being devoted to unpacking these data structures. Further, the expressive power of the data manipulation language was sometimes found lacking, and it became difficult for the application programmer to decide when to use the data manipulation constructs of the data model and when to use those of the programming language.

Our Office Information Model is object-oriented, as defined in [Cardelli 85], since it provides the concepts of objects, classes and class inheritance. As stated previously we believe that office data are most naturally and conveniently represented by organising them into objects - entities that have a persistent and independent existence. Each object is described by a set of attribute values. Objects are classified into classes, and each class has a type, which describes its intension. In addition to attribute descriptions, a type can include the descriptions of operations that are specific to instances of that type. Classes are in turn classified in terms of specialisation/generalisation, defining the inheritance of properties.

Our Office Information Model is strongly typed in the sense that all objects and all expressions must belong to some type and type compatibility is checked for all expressions. The advantages of strong typing in exposing many programming errors are well known. In addition, we believe that classifying objects into types is beneficial in the representation of office data objects, since it encourages data integration, and facilitates the expression of much of the semantics of the application domain in terms of our Office Information Model. Further, we believe that strong typing improves the effectiveness of retrieval since it encourages the user to classify and describe objects more precisely than in a typeless filing facility.

The concept of class inheritance is perhaps the most important characteristic of object-oriented languages [Wegner 87]. It is extremely useful in the

modelling of application domains as it facilitates programs to be specified in a natural and succinct way.

In our Office Information Model, types are organised in a taxonomy of super-type/sub-type relationships. We enforce strict inheritance, i.e. a subtype must possess all the properties of its supertypes. For example, an employee is a subtype, or specialisation, of Person. This means that employees have all the properties of persons, plus the properties particular to employees.

The advantage of this is that all operations that apply to a type can be applied to its subtypes. This can save a great deal of code duplication in application programs. The strict inheritance principle of our type taxonomy is also a very useful means of classifying the data stored in filing and retrieval facilities.

The Office Information Model provides a number of built-in basic types, e.g. integer, real, string and text, together with a number of type constructors e.g. aggregate, list and set. Application programmers can define their own types in terms of the type constructors, the built-in types and other user-defined types, and by defining suitable operations.

Some of the basic data structuring features of our model can be illustrated by the following example of a user-defined type, which is written in an arbitrary syntax and has been somewhat simplified.

```

DECLARE TYPE Report:
  SUPERTYPE: Document;
  ATTRIBUTES: AGGREGATE(
    Title: STRING;
    Authors: SET OF Person;
    Dept: Department;
    Approver: Person;
    Date: AGGREGATE(
      Day: INTEGER;
      Month: INTEGER;
      Year: INTEGER);
    Sections: LIST OF Section)
END;
DECLARE TYPE Section:
  SUPERTYPE: Object;
  ATTRIBUTES: AGGREGATE(
    Title: STRING;
    Content: LIST OF UNION(Section, TEXT))
END;
```

Firstly, this example shows that attributes can be multi-valued, e.g. the Authors of the Report are a set of people. Secondly, the value of an attribute can be an object, e.g. the Authors and Dept attributes. In this way, relationships can be represented. Thirdly, the example shows that hierarchical data structures can be represented directly, e.g. the Date and Sections attributes. Fourthly, recursive data structures can also be represented. E.g. the Sections attribute is a list of objects of type Section; the Section type is recursively defined, since it is a list of elements, each of which may either be of type Text or of type Section itself. Finally, the Section type also illustrates that types can be constructed from other types by applying set operations, in this case Union, to their extensions.

2.2 Additional Features

A very important feature of our Office Information Model is that it facilitates the declaration of powerful constraints. Constraints are necessary because all the information about the properties of objects cannot be expressed as attributes. For example, if we know that the person who approves the issuing of a report may not be one of the authors of that report, then we could attach the following constraint to the type definition given earlier:

```
CONSTRAINT:
  Approver(Report) NOT MEMBER Authors(Report);
```

Another example of the use of constraints is to fulfill the 3rd requirement to our model, concerning the integration of other data models. The background for this requirement is as follows.

In the real world, an Office Information System should be capable of integrating existing tools with new tools, and of storing data that originated outside the system. In general, both these cases lead to the requirement for the Office Information Model to be able to conveniently store foreign data so that they conform to another data model. This is necessary to ensure that programs can be written to translate data represented using another model into our representation so that no information is lost and their semantic integrity is guaranteed, i.e. they can be flawlessly translated back, after being manipulated.

To investigate how to fulfill this requirement, we have looked at one particular case: ISO's Office Document Architecture (ODA) - a standard representation and transfer format for structured office documents. As reported in [Anker-Moeller 87], we discovered that to support ODA, our Office Information Model must support the definition of comprehensive constraints, and provide a meta-type facility.

By a meta-type we mean a type whose instances are themselves types. By a meta-type facility we mean the capability to define new meta-types. Such meta-types can be used to describe the data representation of other data models. It is the constraints specified in these meta-types that ensure that the types that are defined to describe foreign data conform to the required data model.

For example, if we wanted to store data that conforms to the Relational Data Model, then we would declare a meta-type called Relation, whose instances would be the definitions of relations, e.g. Employee, which themselves would be instantiated to tuples representing individual employees.

The provision of constraints in our model leads to a requirement for two other facilities: exceptions and transactions.

The type definitions of our schema presuppose the existence of precise constraints. However, in the real world, which we are trying to model, there will often be some exceptions to these constraints. We therefore provide a facility for application programmers to define exception handlers, which specify what action is to be taken on the raising of an exception. Three basic kinds of actions may be taken:

- the current transaction is aborted
- the current transaction is resumed immediately and violation of the constraint is allowed, e.g. an exceptional value will be stored
- the exception handler executes an operation which deals with the exception, e.g. by returning a permissible value and resuming the transaction.

One advantage of this approach, which is based on the ideas described in [Borgida 85] is that the third course of action allows the programmer to simplify the parts of his program that deal with the normal cases, since the exceptional cases can be "abstracted out".

As indicated above, one reason for having transactions is to allow exceptions to be handled in a manner that maintains the semantic integrity of the stored data. Transactions are also required to cope with recovery in the case of a system crash. Our transaction mechanism supports nested transactions, as defined in [Borgida 85].

Finally, an important feature of our model is the power of user-defined types. Since operations can be associated with an object type, it is possible for programmers to specify object types that are specific to a particular application domain but can be used by all application programmers.

For example, this capability can be used to specify the imprecise data types described in section 3.2. These types will include new definitions of the retrieval operations (EQUALS, LESS THAN, etc.) Similarly, user-defined types can be used to provide versions.

2.3 Conclusions

In this section we have summarized some of the features of the MINSTREL Office Information Model which we have designed to meet the requirements given at the start of the section. Although our model is more complicated than the classical data models, both from a theoretical and implementation point of view, we believe that it can greatly simplify the development and maintenance of advanced, integrated office information systems.

3. Effective Retrieval

3.1 Information Presentation and Browsing

In comparison with a paper-based system, a computerised filing and retrieval system has at least one major disadvantage - difficulty of usage. A computer terminal is a more limited communication medium than paper, mainly due to its small bandwidth and its restriction to two dimensions.

The main aim in the design of the user interface to the MINSTREL system has been to overcome the disadvantages of a computerised office filing and retrieval facility and satisfy the usability requirements for such a facility.

To achieve this we have tried to emulate some of the most useful characteristics of paper-based systems by basing the user interface on the concept of direct manipulation. Direct manipulation is a method of interaction in which the user operates directly on the objects he wishes to manipulate. In other words, our aim is that the user can conceptualise the existence and "location" of information in our computerised system just as concretely as he thinks of the existence and location of paper-documents in a traditional filing system. Further justification, based on field studies, for this "information space" approach is found in [Cole 79]. To fulfill this aim we have made heavy use of graphical presentation, as described below and more fully described in [Hougaard 85].

The user interface to the MINSTREL filing and retrieval facility employs:

- a bit-mapped screen, which allows the display of graphics
- windows, which increase the communication bandwidth

- menus, which generally help to make the system easy to learn and remember
- a mouse, which provides an efficient means of positioning the cursor for direct manipulation.

Each object can be presented in a variety of ways; as a form, as a row in a table, as a relationship diagram, as a data surface, or as an icon. This variety of presentations is necessary to facilitate the presentation of different aspects of an object according to the situation in which it is being presented.

Forms, tables and icons are well-known means of presentation. A relationship diagram is illustrated in Figure 1 below. The object being displayed - the employee "Jones" - is presented as an icon in the centre of the window. All its relationships to other objects are represented by straight lines connecting to icons for the objects related. The lines are labelled with the names of the relationships. The underlying Office Information Model makes this form of presentation possible since it distinguishes objects from simple data values.

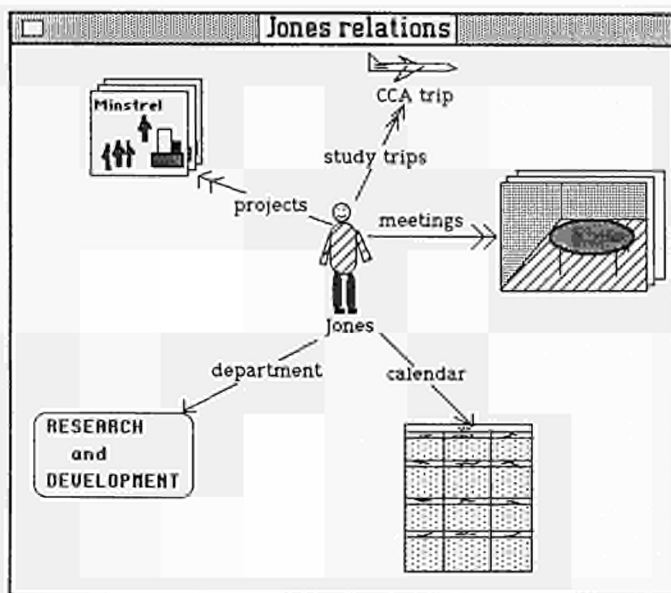


Figure 1: An object presented as a relationship diagram

A relationship diagram is an example of a perceptual encoding, i.e. an encoding in which there is a direct analogy between the concept being conveyed (relationship) and the visual notation used to portray it (connectedness via straight lines). Another example of perceptual encoding is the "projects" icon in Figure 1, which shows that "Jones" works on several projects. Perceptual encoding of the most important information is one of the design principles of our graphical presentations.

To reinforce the user's conception of an information space and to cater for cases in which users cannot specify what they want to retrieve, but are capable of recognising it if they see it, we provide browsing facilities in the MINSTREL system.

Three system capabilities make up the browsing facilities. The first is the capability for the user to select any presentation of an object displayed on the screen, e.g. as part of another object, and display the selected object in a new window in some other presentation form. The second is the variety of presentation forms already mentioned, which aids the user in viewing objects from different aspects. In particular, relationship diagrams encourage browsing using relationships between objects. The third capability is the provision of two classification objects - the type tree and the folder tree. These are special objects which classify the other objects in the system. The presentation of the type tree facilitates browsing in the type classification hierarchy. The presentation of the folder tree facilitates browsing in the hierarchy of folders. Folders are a facility built on top of the Office Information Model that allow the user to group objects in any manner he chooses, i.e. to build up his own classification system. An example of a screen picture with a folder displayed as a data surface and the folder tree in an overlapping window is shown in Figure 2.

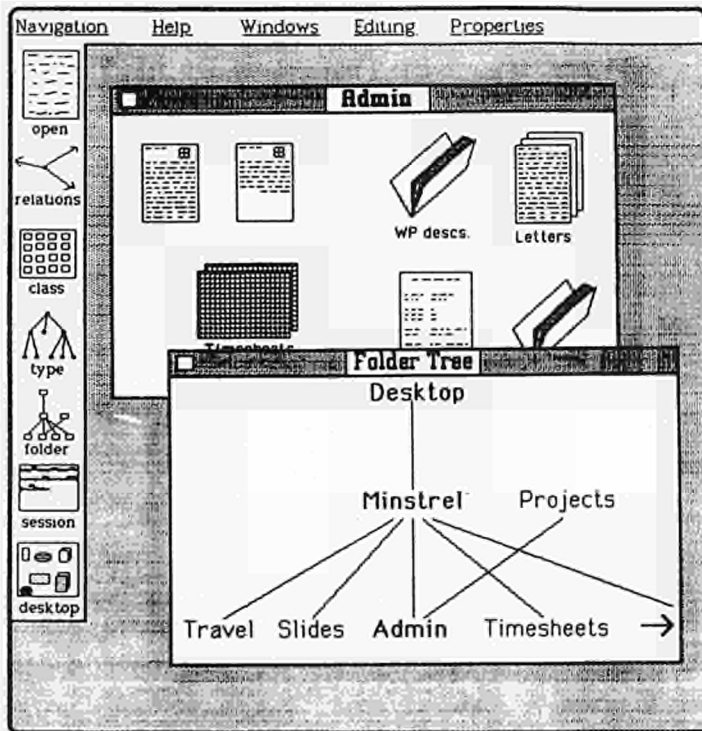


Figure 2: An example of a screen picture during browsing

Acknowledged design principles such as consistency, simplicity and the graphical presentation guidelines described in [Fitter 79] have been employed to try and ensure a usable user interface. Further, thinking-aloud experiments with sample users have been performed to evaluate and revise the design. Our design has demonstrated how careful use of selected graphical techniques,

such as perceptual encoding, windowing and direct manipulation, can be used to alleviate some of the usability problems of computerised office filing and retrieval facilities. A more thorough empirical evaluation of a full-blown prototype system is now required to determine whether we have achieved an acceptable level of usability.

3.2 Query Language

In the past, retrieval of information from computer systems has fallen into two main categories: database management systems and document retrieval systems. The main difference is that document retrieval systems must handle text efficiently.

In our view, office filing and retrieval facilities that are designed to store all office data present several new problems to the design of a query language, since they must fulfill the requirements of both database management and document retrieval systems. The query language must handle both "structured" data, such as customer and part records, and "unstructured" data such as documents and images, in a uniform manner.

The most obvious solution might appear to be the use of boolean logic, a so-called exact match approach, e.g. in a database management system extended to handle text. However, boolean logic has several disadvantages:

- it does not provide very effective retrieval of textual data
- it is hard to use for the typical users of an office filing and retrieval facility.
- it cannot handle imprecise and missing information

In MINSTREL, we have adopted another approach to our query language in order to overcome these disadvantages.

Research within the field of information retrieval [Salton 83] has shown that more effective retrieval of textual data can be achieved using the so-called partial match approach of probabilistic methods. These methods use statistical data about the number of occurrences of terms, e.g. both within individual documents and within the entire collection of documents, to rank documents according to their relevance to the request. In MINSTREL we have chosen a probabilistic text retrieval method, as described in section 3.3, and attempted to extend the partial match approach to satisfactorily deal with the more precise kinds of queries that are usually handled by Database Management Systems.

For the sake of usability, we have attempted to keep the query language as simple as possible. In particular, we have based it on the paradigm of retrieval by instantiation, as proposed in [Williams 82]. We consider the specification of a query to be the same basic process as the creation of an object. To specify a query the user describes the features, i.e. attribute values, of the objects that the query is to retrieve by specifying an archetype (a typical example). The system searches the class of objects of the given type, and for each object, estimates the certainty that it satisfies each of the features. It then adds these certainties to produce a score for each object. The system then returns a list of objects ranked with the highest score first.

Figure 3 shows an example of a query which requests memos which were written by someone from the Sales or Marketing department during the Autumn of 1986, and whose content is about the sales of IBM computers to South Africa and high technology embargoes.

and whose content is about the sales of IBM computers to South Africa and high technology embargoes.

Search Profile			
Memo			
Subobject	Importance	Operator	Sets
From:	Employee		
	Name:		
	Job:		
	Department: Sales / Marketing		
To:	<input type="text"/>		
Date:	860901 - 861130		
Subject:			
Content:	Sales of IBM computers to South Africa. High technology embargo.		

Figure 3: An example of a MINSTREL query

To be able to handle some kinds of more precise, DBMS-style queries, our query language provides the concept of importance. The user can mark each feature as either being ESSENTIAL or DESIRABLE, which is the default. If a feature is specified as being ESSENTIAL, then only objects that satisfy this feature (with some non-zero measure of certainty) will be retrieved.

So for example, if all the attributes in the example of Figure 3 were marked as ESSENTIAL, then it would be equivalent to the boolean query:

```
RETRIEVE Memo SUCH THAT
((Department(From) = "Sales") OR (Department(From) = "Marketing"))
AND (Date >= 860901)
AND (Date <= 861130)
AND (Content ABOUT "Sales of IBM computers to South Africa")
AND (Content ABOUT "High technology embargo")
```

We believe that the provision of importance together with the ability to specify imprecise values, as in the Department and Date attributes of the example, allows users to express a large percentage of their precise queries.

Note that the ability to specify imprecise values fits in perfectly with our paradigm of retrieval by instantiation because we also allow imprecise values to be stored in objects as discussed below. Thus, an object whose values were identical to those of the query specified in Figure 3 might actually be stored in the filing facility.

To deal with any queries that do not fall into this category, we provide set operations (INTERSECTION, UNION and DIFFERENCE). A complex query can be split into several simpler queries, whose results are combined via these set operations. This provides the same power of expression as boolean logic.

Frequently, in Office Information Systems, there is a need to deal with imprecise and missing information. Users will often not know the exact value of one of an object's attributes, and in other cases, some attributes may not even be applicable to a particular object. However, current systems either do not handle such information at all, or they only deal with it in a very limited and not very useful way.

We handle missing information by providing two distinctive null values: not known and not applicable. For example, an object representing an employee who does not have a phone can be assigned the not applicable value for its Phone number attribute, so that it can be distinguished from employees whose phone number is not known.

We handle two types of imprecise information: Firstly, where the user knows that the value is one of a set of possibilities. For example, he doesn't know for certain whether a certain employee is in the Sales or the Marketing department. We call this a P-domain of possible values. The second type is where the value lies within a range of possible values. For example, the user knows that a memo was written some time between 1st September and 30th November last year. We call this a P-range of possible values.

Although this kind of imprecision in a query can be expressed in a boolean query, e.g. by specifying (Date >= 870901) AND (Date <= 871130), our approach has a major advantage in that it can handle missing and imprecise values in the stored data as well as in queries. This is because in our partial match approach we view retrieval as a process of plausible inference [van Rijsbergen 86]. If we view both objects and queries to be sets of statements, then for each object, the system is trying to build up evidence for the proposition that the object implies the query. However, whenever the objects and queries have some text values, there will always be a measure of uncertainty associated with such an implication. Similarly, if there is also some uncertainty associated with the matching of structured attributes, because the objects may have missing or imprecise values, then the plausible inference approach is most appropriate.

Another advantage of our approach is that it can handle other retrieval operations, in addition to ABOUT, which have a measure of uncertainty associated with them, e.g. an APPROXIMATELY EQUALS operation.

We have defined the precise semantics of all retrieval operations (EQUALS, LESS THAN, etc.) in the presence of both imprecise queries and imprecise data. In essence, our solution is to retrieve two sets of objects in response to a query: those which we know are certain to satisfy the query, and those which may possibly satisfy the query. Our approach is unique in that we can quantify the certainty that any object satisfies the query.

We measure this certainty by evaluating the extra information required by the system to be completely certain that an object implies the query. The certainty is inversely proportional to the extra information required. The object about which we need the least extra information is the one we are most certain satisfies the query, and that object about which we need the most extra information is the one about which we are most uncertain.

For example, if we have the following information:

```
colour(object 1) = red
colour(object 2) = [red,orange]
colour(object 3) = [red,orange,blue,green]
colour(object 4) = violet
colour(object 5) = [red,blue]
```

and the query "list all red or orange coloured objects" then clearly object

1 and object 2 satisfy the query with absolute certainty. It is also clear that object 4 does not satisfy the query. But what about object 3 and object 5? We can't say that they satisfy the query nor can we say they don't. We can however say that they possibly satisfy the query and try to estimate the certainty of this statement.

In the above example, we only require one piece of extra information to be completely certain that object 5 implies the query, namely that it is not blue. For object 3, however, we require two extra pieces of information: that it is not blue and that it is not green. Therefore, object 5 would be ranked higher than object 3.

The exact details of how the certainty measures are calculated and a theoretical justification for their applicability will be described in detail in a forthcoming MINSTREL report.

3.3 Text Retrieval

As illustrated for the Content attribute in Figure 3, the user can specify a series of noun phrases (each on a separate line) as the value that a text-valued attribute should be about. In this subsection, we describe our investigation of a suitable retrieval strategy to implement this ABOUT operation, i.e. to return, for each document in the specified class, a measure of the certainty that the value of the specified text attribute is about the specified noun phrases.

As mentioned previously, one of the main advantages of our partial match approach to querying is that it employs statistically-based retrieval strategies for text which have a better effectiveness than the exact match approach. Early on in Project MINSTREL we identified syntactic analysis of natural language as a technique which could be built on top of statistically-based techniques to improve the effectiveness of content retrieval.

We decided to investigate the truth of this hypothesis by implementing some retrieval strategies which use syntactic analysis as part of the retrieval process and comparing the results of applying these with the effectiveness of statistically-based retrieval strategies. Having tried unsuccessfully to obtain a natural language parser to perform the syntactic analysis, our only option was to write our own limited parser, restricting the grammatical coverage to noun phrases. What this meant was that we were not in a position to parse the corpus of texts of our test collection and we were only able to parse queries which would be in noun phrase format. Although this limit on our linguistic processing meant that our hypothesis could not be tried out in full, we still felt that if we could employ parsing of user queries to improve retrieval effectiveness then our ideas would be proved valid.

A large set of experiments which implemented various retrieval strategies were performed on a test collection of 3204 texts and 48 noun phrase queries. All of these strategies parsed the user's query and utilised the resulting information together with the statistical frequency of occurrence and co-occurrence of words from the query and the word order and adjacency of query words in document texts. The details of the experiments will be described in a forthcoming MINSTREL report.

The effectiveness of each retrieval strategy was evaluated by calculating average precision and recall values for the set of queries used [van Rijsbergen 79]. The results showed that our best retrieval strategy gave a small but significant improvement in average retrieval effectiveness. The improvement at the high precision area, which is important in an office filing and retrieval facility, was particularly significant.

From a research point of view, the results we have obtained validate our hypothesis that the syntax of natural language can be used, without the semantics, to improve the effectiveness of this type of retrieval operation. The fact that we have got any improvement in retrieval effectiveness at all, given the crude linguistic processing used on queries only, suggests that further improvements in retrieval effectiveness could be obtained if a better parser (yielding fewer multiple parses and with increased grammatical coverage) was applied to document texts as well as user queries. The approach of using syntactic processing in text retrieval is therefore worthy of further investigation.

If a more sophisticated syntactic processing yielded further improvements in retrieval effectiveness, then it would be worthwhile to include such a text retrieval strategy in future office filing and retrieval facilities. The advantage would be that the documents most relevant to the user's query would be ranked higher in the list of returned objects. In the example, users are more likely to get memos about "sales of computers to the continent of Africa" or the "use of IBM computers in South Africa" rather than memos about "how many IBM computers South Africa sells to Zimbabwe", which is unlikely to be what the user wanted in the first place.

This improvement in the quality of the information systems response is due solely to the surface level interpretation of the user's query. Since the linguistic process used is domain independent, this improvement in quality is applicable to any domain.

4. System Architecture and Prototype Implementation

In MINSTREL we have designed an architecture for an Office Information System that is both flexible and powerful by using the principles of modularisation, layering of functions, and uniform data representation. The architecture that we have used is illustrated in Figure 4.

From the users' point of view, the tools are the most important element of this architecture. A tool is an application program that uses the Object Management System to store its persistent data, and the Dialogue Manager to communicate with the user. A tool provides the user with a particular set of related functions that he can use in performing his tasks within the office. For example, an editing tool can help him in the task of document preparation.

From a system's viewpoint, the kernel of this architecture is the Object Management System, which is the implementation of our Office Information Model, and whose role in an office information system can be compared with that of a Database Management System in an administrative EDP system.

Another major feature of the MINSTREL system architecture is the isolation of all user-system interaction functions in a single system component, called a Dialogue Manager. A final feature of the system architecture is that the Object Management System has been split into two components: the Object Management System and the Storage Management System. The former is responsible for checking the syntax and semantics of the client programs, using the schema information - type intensions and constraint specifications. The latter provides high-level storage management facilities together with low-level access methods, both of which have been specifically designed to provide efficient storage of and access to large volumes of structured and unstructured data.

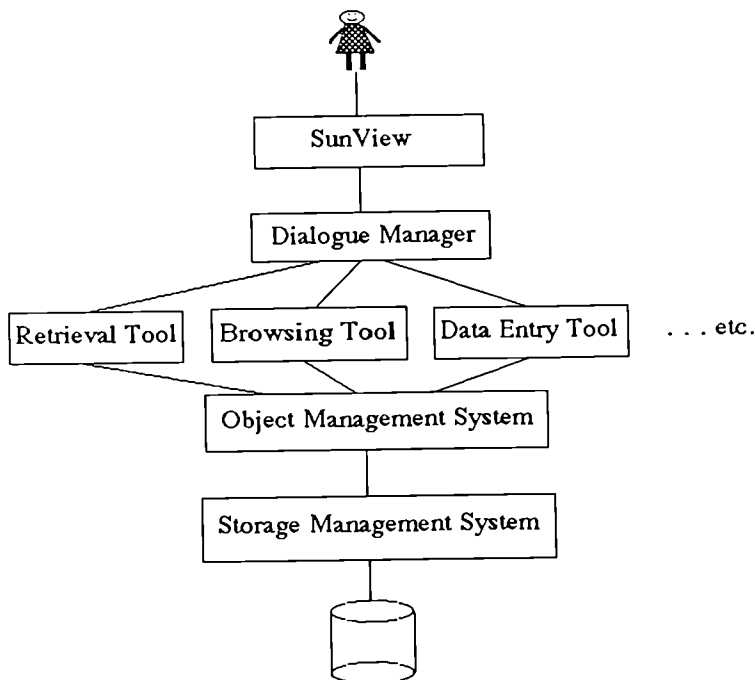


Figure 4: The MINSTREL System Architecture

With the primary purpose of validating our research ideas, we have implemented a prototype office filing and retrieval facility based on this architecture. The prototype is a complete and advanced system with a limited functionality - as we have only implemented a subset of our design. The prototype has been developed using Prolog and 'C' on a SUN workstation. We have implemented a prototype Dialogue Manager, Object Management System and Storage Management System. Using these, we have built Browsing, Retrieval and Data Entry Tools.

The Data Entry Tool is the result of our research into a problem very closely related to computerised filing. Namely, how can existing paper documents be input to the system and represented in terms of the office information model? Most commercial optical character recognisers are incapable of handling the variety and variable quality of paper documents that exist in the real world. We have experimented with some novel techniques which try to overcome this problem, and which will be described in detail in a forthcoming MINSTREL report.

Finally, our prototype also includes a Task Manager, which uses knowledge of the user's tasks to plan and, where possible, automatically execute the invocation of tools to fulfill the user's goals. This research will also be reported on in a forthcoming MINSTREL report.

5. Conclusions

In Project MINSTREL we have investigated some selected techniques that can be used in future office filing and retrieval facilities, which we see as a crucial element of office information systems. Our conclusions are as follows:

- Large-scale, integrated office information systems should be based on a uniform data representation, i.e. an office information model, and a lay-

- ered architecture along the lines described in section 4.
- Such an office information model must provide powerful, high-level representation concepts, including objects, object types, type classification, declarative constraints, transactions, meta types, imprecise data and versions. Outlines of such facilities were given in section 2.
 - Judicious use of graphical techniques, such as those outlined in section 3.1., can significantly improve the usability of an office filing and retrieval facility.
 - The query language for an office filing facility should be based on a partial match approach to retrieval and imprecise queries and data should be handled as outlined in section 3.2.
 - Retrieval effectiveness can be improved by the use of syntactically-based linguistic processing for domain independent systems that store large amounts of textual data, as outlined in section 3.3. And this approach deserves further investigation.

We believe that most of the techniques we have reported on could be advantageously incorporated into commercial office information systems. Once this has been done, the usefulness of the techniques in a production environment can be fully evaluated.

ACKNOWLEDGEMENTS

I would like to thank all the members of Project MINSTREL, both for their inspiration throughout the project and for their vital contribution to the writing of this paper.

REFERENCES

- [Anker-Moeller 87] B. Anker-Moeller, "An Object Type System for a Functional Data Model", DDCl56/1987-03-16/022.
- [Borgida 87] A. Borgida, "Language Features for Flexible Handling of Exceptions in Information Systems" ACM Transactions on Database Systems, Vol. 10, No. 4, December 1985.
- [Brodie 84] M.L. Brodie & D. Ridjanovic, "On the Design and Specification of Database Transactions", in "On Conceptual Modeling", M.L. Brodie et al. (ed.), Springer Verlag 1984.
- [Cardelli 85] L. Cardelli and P. Wegner, "On Understanding Types, Data Abstractions and Polymorphism", Computing Surveys, Vol 17, No. 4, December 1985.
- [Cole 79] I. Cole, "A Structured Interview Survey of Office Filing Systems", HUSAT Memo No. 223, October 1979.
- [Dunnion 85] J. Dunnion et al., "Minstrel-ODM: A Basic Office Data Model", ESPRIT Technical Week 1985.
- [Fitter 79] M. Fitter, "When Do Diagrams Make Good Computer Languages", Int. Journ. of Man-Machine Studies, 1979, Vol. 11, pp. 235-261.
- [Hougaard 85] P. Hougaard and G. McAlpine, "A Graphical User Interface to an Office Filing Facility", ESPRIT Technical Week 1985.
- [van Rijsbergen 79] C.J. van Rijsbergen, "Information Retrieval", 2nd ed., London: Butterworths, 1979.
- [van Rijsbergen 86] C.J. van Rijsbergen, "A Non-classical Logic for Information Retrieval", Computer Journal, Vol. 29, No. 6, December 1986.
- [Salton 83] G. Salton, E. Fox & H. Wu, "Extended Boolean Information Retrieval", Communication of the ACM, November 1983.
- [Shipman 81] D.W. Shipman, "The Functional Data Model and the Data Language Daplex", ACM Transactions on Database Systems, March 1981.

- [Wegner 87] P. Wegner, "The Object-Oriented Classification Paradigm",
Draft, April 1987.
- [Williams 82] M.D. Williams & F.N. Tou, "RABBIT: An Interface for
Database Access", Proc. ACM 1982 Conf., pp. 83-87.

Project No. 385

HUFIT - HUMAN FACTORS IN INFORMATION TECHNOLOGY

K.-P. FÄHRICH, J. ZIEGLER

Fraunhofer Gesellschaft, Institut für Arbeitswirtschaft und Organisation,
Holzgartenstr. 17, 7000 Stuttgart 1
and D.G. DAVIES

HUSAT Research Centre, The Elms, Elms Grove, Loughborough, Leicestershire, UK

ABSTRACT

Fährich, K.-P., Ziegler, J., and Davies, D., G., 1987. HUFIT - Human Factors in Information Technology

The HUFIT (Human Factors in Information Technology) project is an extensive multinational project of cooperation in the area of ergonomics, in the design of IT products, in particular office systems. The project involves eleven institutions in eight European countries. This paper aims to provide a résumé of the work done in the first two years of the project.

1 HUFIT - THE HUMAN FACTORS IN INFORMATION TECHNOLOGY

The preparatory work for HUFIT began in 1982 and 1983. Working for the commission of the European communities, Brian Shackel of the HUSAT Research Centre, Loughborough University, conducted a study on "Ergonomics in Information Technology (IT) in Europe - A review" (Shackel 1984) comparing it to work done in the rest of the world.

The report diagnosed major shortcomings in research and its application in Europe. In parallel to this study, members of the European IT-industry were being secured for partnership in a large multinational project in information technology ergonomics. The Fraunhofer Institut für Arbeitswirtschaft und Organisation (Institute for Industrial Engineering) IAO, Stuttgart, took on the main organisation of this work with the assistance of the HUSAT Research Centre, UK. These two organisations represent the academic centres of excellence. The project began in 1984. The companies who are project partners are Bull (France), ICL (UK), Olivetti (Italy), Philips (The Netherlands) and Siemens (Germany). Also represented in the project as subcontractors to IAO and HUSAT Research Centre are Münster University, West Germany; University College, Cork, Eire; the Piraeus Graduate School of Industrial Studies, Greece and the University of Minho, Portugal. The scientific and technical content of this sizeable research programme is coordinated by the IAO and the HUSAT

Research Centre. The HUFIT project has been established to take on an integrating function for the area "Human Factors and IT-products" within the ESPRIT programme. This means that the project must be closely linked with others and also that it should be opened to more partners at a later stage in its development.

2 MAIN AREAS OF INTEREST IN THE PROJECT

The initial proposition, strengthened by preliminary research, was that appropriately up-to-date human factors knowledge within IT ergonomics should be introduced appropriately into the IT product design process. Equally important is the objective to create the ergonomics methods and tools which will facilitate the development of IT products. It is stated in the project description that: "This project area is concerned with IT-products from the moment of their conception, right through the design and development process, to their installation and use. Its particular contribution is to the development of an integrated human factors input to this whole process." Specific areas of work are:

- analysis of the development processes undergone by IT-products coming from large European manufacturers;
- development of methods for describing task and user characteristics;
- development of methods for implementing usability criteria and evaluation methods in the design process;
- development of a decision-supporting tool for designers;
- gathering human factors knowledge in computer human factors for the decision-support system, and for use by the consortium partners;
- development of human factors tools to assist in IT product design.

The second main area of interest of the project is concerned with the interaction of the user with the system. This involves theoretical and empirical analysis of advanced methods of human-computer-interaction with the intention of creating prototypes of these interaction forms.

Interaction forms under analysis are those consisting of the basic interaction techniques: - "direct graphic manipulation", "natural language" and "formal language", and the work to be done is as follows:

- formal modelling of human-computer-interaction;
- operationalizing the characteristics of different interaction techniques and the definition of generic interaction techniques;
- developing tools for defining and implementing integrated multi-modal user-interfaces, as well as implementing pilot systems;
- theorising evaluation methods.

In the third area of interest, the outcomes from the ongoing project are to be

disseminated amongst interested European IT companies in the form of seminars for managers and designers, workshops, a consultancy service, an industrial affiliates programme and advice to companies or setting up human factors laboratories..

3 PROJECT RESULTS AT PRESENT

In the following, a few important outcomes from the first two years of the project are presented. In addition to from expanding on these results and analysing them further, the primary task of the next phase of the project is to integrate the results into the design of IT products.

3.1 Examination of the IT-Product Development Process

The software design and development cycles as found in the consortium IT companies were put under scrutiny. The goal of the investigation was to ascertain how ergonomics knowledge and practice was employed in today's large IT enterprises. The investigations were carried out by on site interviews and by analysing the design process "guideline" material available from the companies themselves. This dual approach was taken in order to compare the software-engineering intentions of the company the prescribed process with what actually happened in practice, the actual design process.

Olphert et al (1986) extracted a generic model of the product development cycle based on company documentation on product design procedures. The generic design process is intended for the evolution of new products, although in the majority of cases examined, the companies were more likely to be concerned with the development of compatible existing products, in other words, system upgrading.

A field study in the companies (Hannigan & Herring, 1986) showed that, with one or two exceptions, their product development processes had, in reality, relatively little in common with any product design procedures. "The coherent design process into which we can introduce human factors tools does not exist. The process is complex, variable and disorderly--" is among the keynotes to come out of this 'state-of-the-art' study. Little use of explicit human factors or software ergonomics knowledge was found but implicit consideration of human factors was found in three interdependent areas: design guidance, task considerations and usability evaluation. The implicit use of human factors extended beyond planning, development and testing phases into documentation, training, user support and so on.

Some of the designers questioned suggested, amongst other things, that the availability of appropriate software tools would aid the integration of software ergonomics knowledge into the design process. However, in practice things were very different: tools assisting the designer, helping him apply software ergonomics knowledge, were almost untraceable. Where they were available their influence on the design process was negligible. A reason for this state of affairs was often that the

tools available were too complex and did not fit into the design process. Tools or established methods were also non-existent for product evaluation under software ergonomic criteria.

3.2 Designer-Support-Systems in Software Ergonomics

The investigation of design processes in IT companies showed that the classic forms of knowledge transfer (eg. publications, handbooks, design guidelines etc.) have an only limited radius of influence in practice. Another approach is make a computer-based tool allowing integration with other software-development tools available to the IT companies. This idea forms a further part of the HUFIT project, whereby a support-system for product designers is being developed. It consists of four main parts:

- a User-Interface-Management-System for direct manipulative user interfaces;
- a knowledge-based system that has access to several different data and information sources;
- simulation and evaluation modules;
- a conceptual modelling component.

The system is called INTUIT. At the moment the design specifications are being developed (Russell, 1986). The next phase, running to the end of 1987, will see the development of an experimental prototype. Towards the end of 1989 a first, complete, laboratory version of the system should be ready, however, with respect to its ability to function, its integration with other software-engineering tools and its ruggedness, a considerable amount of development work will remain to be done.

3.3 Direct Graphic Manipulation as a Generic Form of Interaction

Bullinger & Fähnrich (1984 b) and Fähnrich & Ziegler (1984) have already formed the hypothesis that direct graphic manipulation can be thought of as a fundamental interaction form. The criticism was also made that, in the literature, this form of interaction has not been clearly enough defined with respect to other interaction forms. This problem has been dealt with over the last two years by HUFIT, whereby a series of systems described as being direct manipulative were investigated. In addition to this, all the relevant literature in this and bordering areas was gathered (Ziegler et al 1985). Three dimensions allowing the characterisation of interaction forms in general, and direct graphic manipulation in particular, were picked out (Ziegler et al 1986 b):

- Representation: The way in which the internal objects of a software system are presented at the visible user interface. In the case of direct manipulation, the external representation is, to a certain extent, an operational model of the application system. This often appears in the form of a metaphor. The user is

able to control the internal objects through manipulation of the surface objects at the interface.

- Referencing: Referencing sets the conditions as to how a user can identify and communicate with objects. The main distinctions are between: pointing actions, giving commands, describing or defining. Direct manipulation interfaces use pointing operations.
- Punctuation: Breaking down the flow of information between user and system into single interaction steps. This dimension sets the complexity and level of interaction of the dialog. In the case of direct manipulation, it is found that dialog steps are quite small and occur with immediate (visual) feedback.

An important step in this context will be to avail these system attributes of commensurate usability attributes, as stated in hypotheses by, amongst others, Bullinger & Fähnrich (1984).

3.4 Aspects of Learning and Cognitive Task Representation

The 'Cognitive Complexity Theory' of Polson and Kieras (Polson & Kieras, 1985) was evaluated and is being continually developed on for the project (Ziegler et al 1986 d). On the strength of the hypothesis that direct manipulation user interfaces are relatively easy to learn (Fähnrich & Ziegler, 1984), the consistency effected by universal (generic) commands was investigated, using the CCT. The hypothesis suggested that great user interface consistency implies great and positive learning transfer between different areas of a system's functionality. A cognitive model of the task, in the form of a production system, was then generated using the CCT, from which it is possible to arrive at quantitative predictions concerning learning times and transfer of learning. Experimental studies (Ziegler et al, 1986 a, b, c) bridging text and graphics editing have shown the CCT to have very high quality of prediction under the desired border conditions. The generic commands under investigation did in fact show good learning transfer, empirically as well as theoretically, on crossing over to the other application area.

Modelling the user's task representation using production rules seems at present to be set at too low a level of abstraction. This forms a key area for further work in the area of CCT. Furthermore, the semantic attributes of the task objects have not been dealt with. This would require improvements in the representation mechanism. Complementary theoretical approaches and methods (e.g. grammar-based approaches) will in the future be examined in greater detail. Up to now the modelling opportunities have been limited to strictly sequentially structured routine tasks, which has made accession to other classes of tasks necessary. On top of this it should not be expected that a balanced picture of the design quality of a user interface, or even a whole product, can be achieved by formal modelling methods alone. Over and above the formal approach, factors important to learning a system are user experience,

knowledge of the area of application and suitable assistance e.g. a 'help' function on the system. A compilation of relevant theories and models was therefore done (Bösser, 1986; Bösser in print). These approaches should lead to a method and tools to determine learning requirements and also a possible learning supports.

3.5 Formal modelling and Description of Human-Computer-Interaction

The project has submitted a study comparing formal methods of modelling in the area of human-computer-interaction (Hoppe et al, 1986; Hoppe, 1986), whereby two main areas within the methods were investigated. The one dealt with cognitive oriented modelling processes which are used as analytic tools for task analysis and description, and which offer a way of predicting user performance, learning behaviour and knowledge transfer. These analytic tools are in part extensively formalized, allowing them to be implemented on a computer, thus facilitating quantitative measurement. The other involved the investigation of formalisations and models suited to well-specified user interface and to the automatic production of a functional system. These approaches are particularly important for the development of User-Interface-Management-Systems which can be used for rapid-prototyping interfaces, more specifically, direct manipulation interfaces. Methods which can be classified under state-transition networks as well as grammatical description are important approaches. Many of these methods are at the moment being used in the project for implementation tasks, thus being further tested for their suitability. For example, state-transition representation is being applied in a PROLOG environment for rapid prototyping of user interfaces comprising of linguistic and textual components. A future goal in this area will be to bring together the approaches generating user interfaces and the cognitive-oriented, predictive process, and, during system development, to make available not only implementation tools but also support systems for assessing the various implementation alternatives.

3.6 Classification System

Part of the HUFIT project is the GLOT (Glossary of Terms) exercise. This concerns the production of a multilingual glossary for the area "Information and Communication Technology". The glossary, with c. 2000 terms and definitions has been submitted (Hoepelman et al, 1986) and is currently being evaluated by external experts. Central to the project is a subsection involving the production of a glossary of software-ergonomics.).

The collation and classification of computer human factors (CHF) knowledge is an essential component of the HUFIT project. Primary and secondary journals within the field of CHF from 1982 onwards, and material from specialist conferences, commercial abstracting and indexing services are regularly searched and relevant material is added to the data base of bibliographic references. Access to the bibliographic data

base is via a classification scheme and a thesaurus. The classification scheme and thesaurus have been developed using the user-centred design approach. The human factors experts in the partners and elsewhere were the source of the classification scheme via workshop. Pilot literature searches on Usability, Learnability, and Design Processes have been conducted for the partners to assess the usefulness and ease of use on the data base. In the future the data base will be transferred onto a computer. Currently it is mainly a paper resource.

3.7 User/Task Analysis and Classification

In order that products can be designed to meet the requirements of the user to carry out tasks, better understanding of the user and the task are required at various stages in the design process. Work in this area so far has renewed the literature on task and user analysis. The general outcome is that although a number of methodologies exist they are primarily research tools, unsuitable for use in the design process. The problems associated with the use of these methods include being extremely time consuming and not providing information in a form which is usable by designers. The current work is concerned with the methodological difficulties associated with ergonomic design of off-the-shelf products. This is taken further by analysing the ergonomics action constraints on three main types of design: enhancement of existing products, design of new products, buying in an OEM. This leads to a framework for the analysis of tasks and users. The main concern about these methods in their integration into the structure of an IT product manufacturer, hence it is essential to deal not only with data for design. Further work will be applied research, aiming both at organisational changes within companies and at clarification of methodological issues.

3.8 Usability in the Design Process.

From the systems design point of view usability research has to fulfil two needs; the identification of the criteria which indicate usability, and the development of techniques aid systems designers to use these criteria in the design process and to evaluate wether design solutions are usable. An extensive review of the literature has been undertaken and has shown that usability criteria are difficult to identify and to quantify. Current views suggest that designing for usability requires that usability is defined in the product specification, that usability goals are included in design guidance and products are tested for usability. This implies a certain type of design process. The fundamental features of such a design process are that it should be:

- user centred
- participative
- experimental
- iterative

- user-supportive

The project is currently investigating ways of fulfilling these requirements.

4. CONCLUSIONS

The HUFIT project is an ambitious multi-national research and implementation activity involving two University based centres of excellence, five European IT companies and four University subcontractors. The key objective of the project is to provide the European IT industry with means for developing products which are more closely matched to the needs, requirements and characteristics of users. It takes as its primary assumption the view that the incorporating of human factors into IT product design is an essential pre-requisite to the effective use and acceptance of future generations of office systems.

5 REFERENCES

- Bösser, Tom (in print): Learning in Man- Computer Interaction. A critical review of the literature. In: Springer Lecture Notes.
- Bösser, Tom (1986): Modelling of skilled behaviour and learning. In: Proceedings of the IEEE Conference on Systems, Man and Cybernetics (Atlanta, Georgia, October 14-17, 1986). New York: IEEE, pp. 272-276.
- Bullinger, H.-J. & Fähnrich, K.-P. (1984a): Software-Ergonomie-Konferenz-berichte. Interner Report des Fraunhofer-Instituts für Arbeitswirtschaft und Organisation (IAO), Stuttgart.
- Bullinger, H.-J. & Fähnrich, K.-P. (1984b): Symbiotic Man-Computer-Interfaces and the User Assistant Concept. In: Salvendy, G. (Hrsg.): Interact '84 Proc. of the First USA-Japan Conference on Human-Computer-Interaction, Honolulu, Hawaii, August 18-20, 1984, ELSEVIER, Amsterdam, New York, Oxford, Tokio, pp. 17-20.
- Bullinger, H.J., Davies, D.G., Fähnrich, K.-P., Shackel, B., Ziegler, J. (1986): Research Needs and European Collaboration in Human-Computer-Interaction. In: Proc. "Work with Display Units", Stockholm, 1986.
- Davies, D.G. (1986): HUFIT's Role in Office Systems Design. ESPRIT '86 North-Holland, Amsterdam, New York, Oxford, Tokio, 1986.
- Gaines, B. (1984): From Ergonomics to the Fifth Generation. 30 Years of Human-Computer-Interaction-Studies. In: Shackel, B. (Hrsg.): Interact '84: Proc. of the First IFIP Conference on Human-Computer-Interaction. Volume 1, p. 1.1, London.
- Fähnrich, K.-P. & Ziegler, J. (1984): Workstations Using Direct Manipulation as Interaction Mode. In: Shackel, B. (Hrsg.): Interact '84: Proc. of the First IFIP Conference on Human-Computer-Interaction, London.
- Fähnrich, K.-P. (1985): European Human-Factors Laboratory in Information Technology. In: Bullinger H.-J. (Hrsg.): Proc. "Human Factors in Manufacturing", IFS Publications. UK, 1985.
- Hannigan, S., Herring, V. (1986): The Role of Human Factors Inputs to Design Cycles; Deliverable A1.2b, HUFIT CODE: HUFIT/8-HUS-11/86.
- Hoepelman, J., Heller, N., Thiele, S. (1986): Revised Draft of Glossary of Terms; Working Paper C6.3, HUFIT CODE: HUFIT/9-IAO-6/86.
- Hoppe, H.U., Tauber, M., Ziegler, J. (1986): A Survey of Models and Formal Description Methods in HCI with Example Applications; Deliverable B 3.2a, HUFIT CODE: HUFIT/12-IAO-7/86.
- Hoppe, H.U. (1986): Cognitive Modelling - A New Tool for User Interface Design and Evaluation. In: Proc. "AI Europe", Wiesbaden, 1986.
- Olphert, W., Galer, M.D., Hannigan, S., Russel, A.J. (1986): Design Cycle Model; HUFIT Working Paper A1.1b, HUFIT CODE: HUFIT/3-HUS-3/86.

- Phillips, K.E. (1985): Classification of Domains of Human Factors in Information Technology; Working Paper A3.1.a, HUFIT CODE: HUFIT/1-HUS-5/85.
- Phillips, K.E., Galer, M.D. (1986): The Development of a Computer Human Factors Classification and Collation of Human Factors Knowledge; Interim Report A3.1, A3.2; HUFIT CODE: HUFIT/4-HUS-8/86.
- Phillips, K.E. (1986): A Pilot Search on the Computer Human Factors Data Base, Topic: Usability; HUFIT CODE: HUFIT/5-HUS-7/86.
- Polson, P., Kieras, D. (1985): An Approach to the Formal Analysis of User Complexity; *Int. Journal Man-Machine Studies*, Vol. 22, 365-394.
- Russell, A.J. (1986): Knowledge Base Structure and System Specification; Working Paper A2, HUFIT CODE: HUFIT/2-ICL-01/86.
- Shackel, B. (1984): Ergonomics in Information Technology in Europe - A Review. HUSAT Memo No. 309. Report für die Kommission der Europäischen Gemeinschaften, 1984.
- Ziegler, J., Vossen P.H., Hoppe H.U., Fähnrich, K.-P. (1985): Analysis of Direct Manipulation Interfaces, Part I and Part II; Working Paper B3.1a, HUFIT CODE: HUFIT/4-IAO-12/85 und HUFIT/5-IAO-12/85.
- Ziegler, J. (1986): Analyse kognitiver Aufgaben in der Software-Ergonomie. Gesellschaft für Informatik, In: Proc. "Software-Ergonomie Herbstschule"; Berlin, Oktober 1986.
- Ziegler, J., Vossen P.H., Hoppe H.U. (1986a): Cognitive Complexity of Human - Computer Interaction. In: Proc. "ESPRIT '86" Noth-Holland, Amsterdam, New York, Oxford, Tokio, 1986..
- Ziegler, J., Hoppe, H.U., Fähnrich, K.-P. (1986b): Learning and Transfer for Text and Graphics Editing with a Direct Manipulation Interface. In: Proc. "CHI '86, Computer - Human Interaction"; Boston, April 13-17, 1986.
- Ziegler, J., Vossen P.H., Hoppe H.U. (1986c): On Using Production Systems for Cognitive Task Analysis and Prediction of Transfer of Skill. In: Proc. "3rd European Conference on Cognitive Ergonomics", Paris, Sept. 15-19, 1986.
- Ziegler, J., Eichhorn, R., Hoppe, H.U., Puetz, R. (1986d): Direct Manipulation in a General Framework of Human-Computer Interaction; Working Paper B3.1b, HUFIT CODE: HUFIT/7-IAO-6/86.

ACKNOWLEDGEMENTS

The authors would like to thank their colleagues in the ESPRIT-HUFIT consortium for their contributions to the work cited and the successful progress of the project.

HUMAN FACTORS ENGINEERING OF INTERFACES FOR SPEECH AND TEXT IN AN OFFICE ENVIRONMENT

F.L. van Nes

Institute for Perception Research/IPO
P.O. Box 513
5600 MB Eindhoven, The Netherlands

Today's data-processing equipment almost exclusively uses one input medium: the keyboard, and one output medium: the visual display unit. An alternative to typing would be welcome in view of the effort needed to become proficient in typing; speech may provide this alternative if a proper speech recognition interface is available. As to the output, visual data presentation through text and graphics is limited by physical display properties. Visual data can be supported by auditory data, provided the human factors aspects of both presentation modes are taken into account. This paper presents results of two experiments on the ergonomics of voice input and output. The first deals with HELP messages in speech or text form, and shows that speech is a good medium for such messages, especially for learning the operations required. The second experiment compares spoken with typewritten annotations to a text. Making voice annotations seems most efficient, since longer annotations are produced in less time.

1. INTRODUCTION

At the moment, data processing equipment such as computers or information systems almost exclusively use one input and one output medium. Input is predominantly generated by the user by means of a keyboard or keypad. Output is produced as text on a visual display, mostly a CRT or a printer, and thus has to be read.

As to the input side, conventional (QWERTY) keyboards can in principle be operated by almost everybody, but to become reasonably proficient in typing, in terms of speed and accuracy, quite a learning effort is necessary. Without possessing this skill, typing is slow and error-prone, as becomes apparent when watching the majority of occasional typists, e.g. computer operators. An alternative to typing would therefore be welcome, provided its use is advantageous (Ogozalek & Van Praag, 1986). Speech may present this alternative in the near future, since speech recognition technology has made considerable progress. However, data on the ergonomic aspects of speech input is scarce, to say the least. Knowledge in this area must therefore be gathered through experiments like the one reported below.

As to the output side, one need only look at the cluttered screens in data processing environments to realize the limits of displaying information through text and graphics alone (Van Nes, 1986). Fortunately, visual data presentation can be supported by auditory presentation which, moreover, need not be restricted to whistles and bleeps but may be in spoken form as well (Van Nes, 1982). Research is then needed to ensure that the addition of voice data supports visual data presentation. In some cases, for instance the special services that can be provided in telephony, auditory output is the only possibility; again, the ergonomic aspects of voice output in such situations should be investigated in experiments (Aucella & Ehrlich, 1986).

This paper presents results of some experiments on voice input and output, performed in the ESPRIT-HUFIT project no. 385, "Office automation". The experiments are part of our efforts to produce data to enable appropriate human-factors engineering of voice and text interfaces. Therefore, tasks typical of an office environment were investigated. Office tasks provide interesting and, from a human factors point of view important, examples of applications of data processing technology for professional users without, however, special training in data processing.

2. USE OF SPEECH FOR CONTROL AND CONTENT INFORMATION

In the dialogue between an information system and its user, two types of information may be distinguished: "control information" which is necessary to enable the interchange of data to take place between man and machine, and the "content information" that is represented by these data. Generally, the user does not want to be bothered more than is absolutely necessary with the control information; the content information is what he cares about.

Control information consists of fixed, constant messages. Examples are: commands from the user, system messages and instructional information. The latter may be made available on request by the user, as in a HELP system. Such a situation might obtain for professional users of systems who are not data processing specialists. For instance, secretaries or managers using a word processor and needing to perform a text processing function that they have seldom or never used, have to consult a manual if the word processor itself cannot provide the information required. If it does contain such information, however, it is of interest to investigate whether this should be offered in visual or auditory form, i.e. as text or speech. There are promising examples now of the latter possibility (Nakatani et al., 1986).

Content information consists of variable messages that are wholly determined by the user and the application concerned. Examples are a document that is produced on a word processor, and comments or annotations to this document which are made by someone other than the author. Traditionally, i.e. for printed documents, the annotations are produced in written form since they usually cannot be generated on the spot, while the document author is present. However, voice may be a better medium than script for giving such comments, especially when they are lengthy or subtle. The availability of "voice store and forward"-technology now makes it possible to listen to voice annotations at another time and place than when and where they were recorded. It is therefore of practical interest to investigate whether voice or script annotation is more suitable for the aims and needs of the "sender" and the "receiver" of such annotations.

Two experiments will be described below on the use of voice for control and content information, respectively. The first experiment is concerned with speech output, the second one primarily with speech input.

3. EXPERIMENT I: SPEECH OUTPUT VS. TEXT OUTPUT FOR HELP INFORMATION

3.1. Introduction

An inquiry among 200 users of a particular word processor showed that some of its 69 command functions were not known by heart to most users, although these "unknown" commands were employed from time to time. Such commands seem to be both difficult and important, and therefore would be prime candidates for a HELP function implemented on a word processor able to give instructions in oral or written form.

3.2. Method

A small experiment was done with eight novice users of the word processor mentioned, i.e. six of them had never and two hardly ever used it. They could obtain instructions from a separate system, with a keypad as input device and either a visual display or loudspeaker as output device. The latter provided PCM-coded speech output from a stand-alone minicomputer. All subjects performed two equivalent tasks, each consisting of three parts; all involved the changing of margins in an existing text. Half of the subjects started with task 1, with instructions in the speech mode, then continued with task 2, with instructions in the text mode. The other half started with task 1 with text instructions, and continued with task 2 with speech instructions. The spoken HELP messages could be stopped as well as repeated from the start.

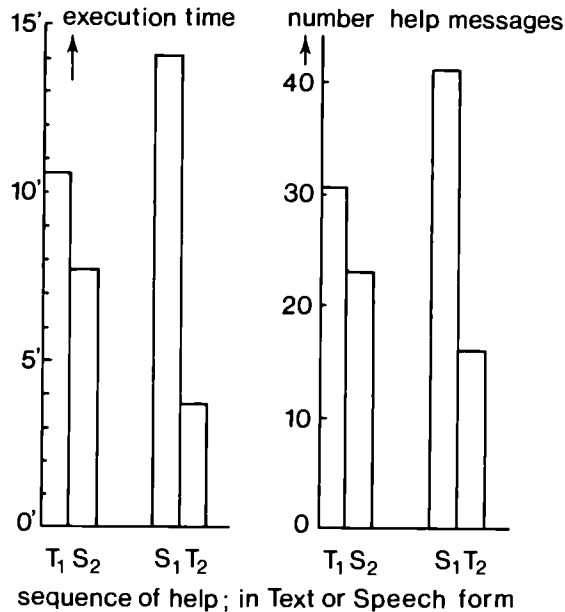


FIGURE 1
Average task execution times in minutes (left) and requested number of help messages (right). T₁ and T₂ refer to Text help in task 1 or task 2, respectively; S₁ and S₂ refer to Speech help in task 1 or task 2, respectively.

3.3. Results: performance

Two main variables were used to measure performance: task execution time and requested number of HELP messages, for task 1 and task 2. These results, depicted in Figure 1, clearly show a learning effect from task 1 to task 2. The extent of the improvements in overall execution time and requested HELP messages, however, depends on the order in which the subjects received spoken or written instructions: the mean execution time decrease for speech first was 74%, that for text first 27%, whereas the mean decrease in requests for HELP

was 61% for speech first against 25% for text first.

3.4. Results: preference

At the end of the two tasks the instructor asked the subject which he/she preferred, the text version or the speech version, and the reason(s) for their preference. Five subjects preferred speech and the other three subjects preferred text as the output medium of the HELP system.

The reasons given for preferring the speech mode were:

- . the possibility of obeying the command simultaneously with the spoken instruction. An instruction was divided into a maximum of four steps, i.e. key presses; in the speech mode, it was possible to look at the keyboard and press the required key while listening;
- . the fact that prosodic factors helped the user to grasp the most important information;
- . the "human character" of this type of communication;
- . the more relaxed way of perceiving information by speech than by text.

The reasons given for preferring the text mode were:

- . the random accessibility of specific fragments;
- . the permanent availability ("you don't have to remember what to do when the instruction is given in text", one subject remarked).

4. EXPERIMENT II: VOICE VS. SCRIPT ANNOTATION

4.1. Introduction

In the office, workstations with such facilities as word processing, voice annotation and electronic mail have already appeared or will do so in the near future. Annotating a document and reflecting on these annotations involves the integration of two information sources, in one or even in two sensory modalities, depending on the use of script- or voice annotations, respectively. Probably the course of this integration is different for producing versus receiving annotations, therefore both should be studied experimentally. We have started with investigating the process of producing or "sending" annotations, in typewritten and spoken form.

4.2. Method

An experiment was done with twelve subjects who were asked to annotate four screen pages of text that were divided in two pairs, that were presented on a high-resolution flicker-free display of a SUN 2 workstation allowing the presentation of dark text on a light background, with various fonts and fine-detail graphics. On such a high-quality display a printed-paper-like document can be presented, which seems a favourable setting for a realistic annotation task. Script annotations could be made in display windows next to the text being annotated. Voice annotations were taped in analog form on a digitally controlled recorder. Half of the subjects annotated the first pair of pages with script, the second pair with voice; the other subjects used the two annotation modes in reverse order. The order of the two pairs of pages was also counterbalanced over subjects and annotation modalities.

4.3. Results: performance

Nine voice annotations were made on the average per subject, with a mean length of 13.8 words, and eight script annotations, with a mean length of 7.5 words. Roughly speaking, the number of annotations was the same in either mode, and voice annotations contained twice as many words as script annotations. The number of voice annotations ranged from three to nineteen for individual subjects; the number of script annotations from two to fourteen. The numbers of

voice and script annotations each subject made were significantly correlated.

The annotations were divided into four classes according to content:

- . objective remarks about a fact in the text;
- . remarks about a fact in the text which expressed an opinion of the annotator;
- . remarks about a typing error;
- . remarks about the writing style of the author of the text.

There were no significant differences in relative numbers of voice and script annotations in any class.

It is of interest to compare the times the subjects needed for their spoken or typewritten annotations, although these times not only reflect the annotation act per se, but the respective interfaces for voice and script as well. Table 1 shows the average total times subjects needed for their complete annotation tasks, as well as the times spent on reading and thinking about the text and then on annotating it.

TABLE 1
Mean total task execution, reading and annotating times (in minutes and seconds), with their standard deviations between brackets, for the voice and the script annotation system.

	annotation system	
	voice	script
Mean reading time	8:21(5:05)	11:18(7:34)
Mean annotating time	7:45(4:14)	20:34(12:16)
Mean task time	16:06(8:43)	31:52(17:59)

4.4. Results: preference

Immediately after the annotation tasks subjects were asked which system they preferred, and why. Eight subjects preferred the voice, and four the script annotation system, for a variety of reasons. Those preferring voice mentioned, for example:

- . it is faster;
- . less reading is necessary (i.e. no typewritten annotations have to be checked);
- . complete sentences are used more easily.

Those preferring script said, for example:

- . it is easier to think about the annotation content;
- . it is easier to make changes;
- . it is easier to type than to speak after reading the document because you "stay in the same mental framework".

4.5. Conclusion of the annotation experiment

To sum up, making voice annotations seems more efficient: longer annotations (albeit with virtually the same amount of information) were produced in less time.

5. DISCUSSION AND CONCLUSIONS

The reported results show that speech can be a valuable medium, as good as text or better, for presenting control and content information-output, and generating content information-input. The application areas for speech should be carefully selected so as to take advantage of its favourable properties and try to compensate for its weak points. Training may be such an application in view of the fact that, in Experiment I, both task execution time and number of evoked HELP messages might be considered to show a larger positive transfer of training for the subjects who first used spoken HELP messages than for those who first used written HELP. This is in accordance with a finding by Potosnak and Van Nes (1984). However, the volatility and low random accessibility of spoken messages present problems. As to accessibility, in a follow-up of Experiment I, a "pause" facility was introduced in addition to the facilities for "stop" and "repeat", making it possible to stop and then continue the voice message. This proved to be advantageous, but not wholly satisfactory yet for accessing a desired part of the message.

Another application area for voice will certainly be annotating and commenting. Especially for long, complex comments, as might be elicited when commenting on a scientific manuscript, voice seems eminently suited. However, the described experiment dealt with the process of making annotations: receiving and dealing with previously unknown annotations should be studied as well.

ACKNOWLEDGEMENTS

The work reported in this paper was done by a team of which all members contributed to the results. Many thanks are extended to Jan Douma, Joe Hary, Theo de Jong, all the subjects, and especially to Piet van Lingen and Elly van Veghel, who did the annotation and word processing experiments, respectively.

REFERENCES

- Aucella, A.F. & Ehrlich, S.F. (1986) Voice messaging enhancing the user interface based on field performance. In: Proceedings CHI '86 Human Factors in Computing Systems (Boston, April 13-17, 1986). New York: ACM, 156-161.
- Nakatani, L.H., Egan, D.E., Ruedisueli, L.W., Hawley, P.M. & Lewart, D.K. (1986) TNT: a talking tutor "N" trainer for teaching the use of interactive computer systems. In: Proceedings CHI '86 Human Factors in Computing Systems (Boston, April 13-17, 1986). New York: ACM, 29-34.
- Nes, F.L. van (1982) Perceptive, cognitive and communicative aspects of data processing equipment. In: Proceedings 1982 International Zürich Seminar on Digital Communications - Man-Machine Interaction (Zürich, March 9-11, 1982). IEEE Catalog no. 82 CH 1735-0, Zürich, 259-262.
- Nes, F.L. van (1986) Space, colour and typography on visual display terminals. *Behaviour & Information Technology*, 5 (2), 99-118.
- Ogozalek, V.Z. & Praag, J. van (1986) Comparison of older and younger users on keyboard and voice input computer-based composition tasks. In: Proceedings CHI '86 Human Factors in Computing Systems (Boston, April 13-17, 1986). New York: ACM, 205-211.
- Potosnak, K.M. & Nes, F.L. van (1984) Effects of replacing text with speech output in an electronic mail application. IPD Annual Progress Report, 19, 123-129.

The INCA Workstation

C. Bathe; H.P. Godbersen
Nixdorf NME
Berliner Strasse 66, D-1000 Berlin 27

Summary

Two subprojects under INCA , a multimedia workstation and a broadband LAN, started under pilot project 95 in 1984. The plans for both these developments were presented in the 1985 ESPRIT technical week. Now, two years later, the two developments are essentially complete. This paper describes the realization of the workstation activity. In particular, the Document Preparation application, the Communication Support, and the hardware development of a high resolution display are discussed. Finally an outlook for future developments is given.

1. Introduction

In Project 395 (*INCA: An Integrated Network Architecture for Office Communications*) we have defined a *client/server concept* with *Workstations (WS)* and *dedicated servers* (Portion Server, Mail Server, etc.), connected by a Broadband System which offers a set of 2 Mbit/s channels, accessed by *Network Interface Module* with *agile modems*, thus providing an aggregate throughput of some tens of MBit/s. We have also defined a set of communication protocols offering both *stream and packet mode services*. Figure 1 outlines the local client/server scenario. These typical local configurations will be connected by PSDN, ISDN, or PSTN. The main task of INCA is to define a communication architecture for the office environment, decide on protocols, and to demonstrate the results with realistic applications. It should be noted, that our applications are designed to run in the general INCA scenario at the partner's sites [7]. However, in the following we will concentrate on a local configuration, in which the broadband approach may be even substituted by an ordinary LAN.

The subproject combines the *technology driven* with the *application driven* approach, by checking the transport system characteristics against the requirements of future office applications, particularly in the field of *Document Preparation* of multimedia documents (in-

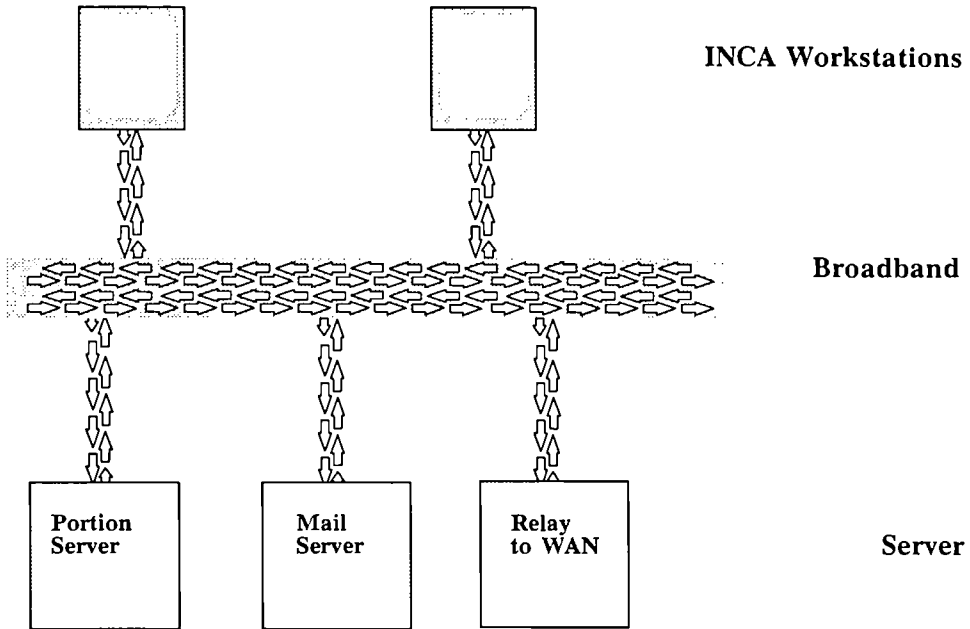


Fig. 1: The INCA Client/Server Scenario

cluding text, image, graphics, etc.) and *Electronic Mail*. This approach allows to run the different INCA networks with realistic loads.

Figure 2 depicts the upper protocol layers and application modules that reside in a typical INCA WS. The two major applications are *Document Preparation (DP)* and the *Mailbox Access Unit (MAU)*, both embedded in an Application Supervisor, that interconnects the applications and the Human Interface. DP makes use of existing subeditors (e.g. for graphics). A *Document Portion Manager (DPM)* closes the gap between the document orientation of the applications and the file orientation of the servers. It makes use of standard Archive-, User- and Network Management Services. The MAU communicates with the Mail Server by means of the P7 protocol. The *Mailbox Client Entity* maps P7 to the *Remote Operation Service (ROS)*. A more detailed discussion of the design concept can be found in [1].

The next section discusses the DPM including PT. The applications are looked at in section three. Chapter four features the hardware development. Finally an outlook is given.

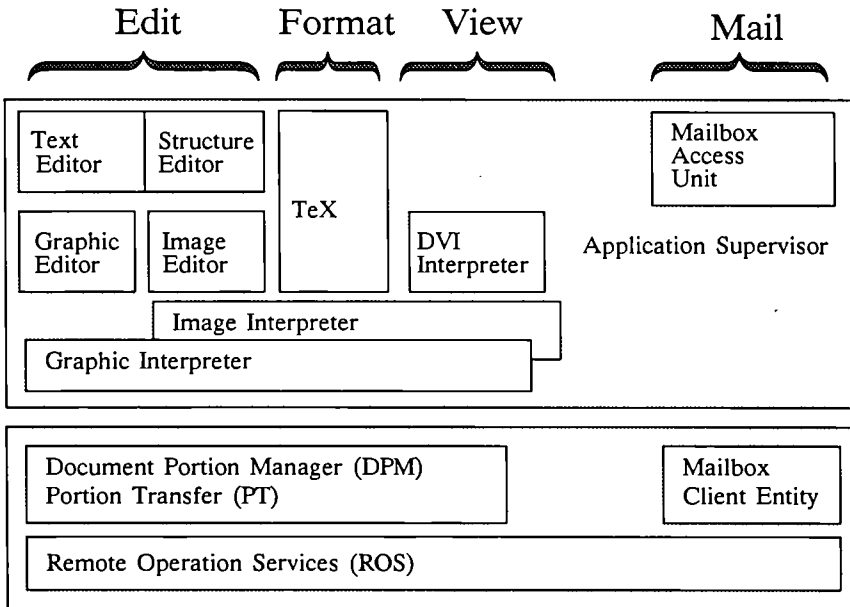


Fig. 2: The Main Components of the INCA Workstation

2. The Document Portion Manager

The Document Portion Manager (DPM) acts as an interface between the applications the communication subsystem. The main task of the DPM is the management of document portions, their retrieval and storage, and the handling of informations contained in the header of a document by making use of services provided by the Portion Transfer (PT). The general principles of the DPM design are:

- *Document oriented user interface:* The only objects the DPM supports, are documents and parts of them. The DPM hides where a document is stored, and how it is stored on file servers. It hides the multi channel communications environment of the underlying broadband network from the application.
- Each document consists of a document header and zero or several document portions. Multiple versions of the same document are regarded as single documents, i.e. each version of a specific document is represented by a document header of its own.
- Document portions may be shared among more than one document. This is typically the case when mailing a document from one user to an other; only the docu-

ment header is duplicated and mailed which results in two document headers containing reference information to the same document portions.

- * *Application independence:* The DPM services are not restricted to a special document type (e.g. SGML, ODA), but supports a quite general class of documents. The Document Header (DH) contains all information about the document. A document is of a certain document type, which specifies which portions are allowed. Each portion is of a certain portion type. To support application independence, both DPM and Application share a list of all available portions for each document operated upon. Currently the following Portion Types are supported:

SGML Generic Structure	.gen
SGML Specific Structure	.spe
Text	.txt
Image	.img
Image Delta File	.dlt
Graphic	.gra
Graphic Metafile	.cgm
TeX File	.tex
DVI File	.dvi

The second row indicates the suffix of a temp file name, under which these portions are made available to the applications (cf. section 3).

- * *Multi document, single user operation:* Several documents may be handled simultaneously by a DPM, but it will allow only a single user at a time. It is supposed that the single user assumption is not a real restriction since an editor will always work for a single user.
- * *Multiple access modes:* The DPM Services will support three access modes for working on a document: *read, write, and update*. The access mode and the concurrency control apply to all portions belonging to the document.

Figure 3 gives an overview of the components and interfaces of the DPM. The Application Supervisor interacts with the DPM via a command interface: The command `D_Bind` invokes a new DPM process. The DPM process terminates with a `D_Unbind`. The following services relate to an entire document: `D_Create`, `D_Delete`, `D_Open`, `D_Close`, `D_Copy`, `D_Read_Attributes`, `D_Change_Attributes`, `D_Prepare_Mail` and `D_Integrate_Mail`. Typical attributes of a document are author, title, creation date, etc. The two mail related commands are provided to support our concept of sending only document headers by mail (cf. [1]). Additional services are provided for a specific Document Portion: `P_Create`, `P_Delete`, `P_Get`, `P_Put`, and `P_Copy`.

The DPM provides the Supervisor with different database tables and temporary files. Four different Database Tables are managed by the DPM: *User/Network* contains mappings from Users and Document types to specific servers. *Archive* holds a list of Document titles. The *Document Table* holds the attributes of the documents currently opened by the user. The names and the status of the portions for each of these opened documents is made avail-

Portion Transfer

The Portion Transfer (PT) provides its services to the DPM and interconnects this application entity to functional components located within the file servers. The following interface is provided: `PT_clfi()` closes a file on a server. `PT_cofi()` copies a file inside a server. `PT_cone()` connects to a server. `PT_crfi()` creates a file on a server. `PT_defi()` deletes a file on a server. `PT_gefi()` reads a file from a server into a tempfile. `PT_opfi()` opens a file on a server and controls the access rights. `PT_pufi()` puts content from a tempfile into an open file. `PT_rele()` disconnects a server. The actual file transfer to the servers is currently realized by TCP/IP.

3. Document Preparation

One of the design principles was to maintain *independence of software components*, and to make extensive use of existing software. The Application Supervisor serves as a glue, it is implemented with the help of a program generator. Figure 4 depicts the typical process of preparing a document (cf. [2]). The structure editor (SE), based on SGML, aides the user to create a specific logical structure of a document. The text content is created by means of an existing text editor (TE). Both editors are linked together as a single process. The formatting process is initiated by filtering the result of the editing process towards TeX. The viewing process runs under the supervision of the DVI interpreter. The Graphic and Image Interpreters are called on request. The editing of images and graphics is separated. All the files produced in this document preparation process can be stored as portions of a document under the control of the DPM.

Interface between Text- and Structure Editor

Our goal in defining an interface between SE and TE was to reduce changes in the TE software to a minimum. The main task of the SE is to manipulate the logical tree structure (e.g. create, delete, move, and copy objects). Changes introduced by the SE have an impact on the actual text content (which remains under control of TE). The link between the nodes of the logical structure tree and the actual text content is created by references. For example, a paragraph of text is identified by two marks. If the user decides to delete this paragraph, the TE can identify and delete its content by means of the reference mechanism.

The following functions control the assignment of content references and TE operations resulting from SE demands. Function `Stpjck()` saves the actual object. If content portions are associated, the TE will be called to save it. `Stcut()` will unlink the actual logical object and its attribute descriptions out of the logical tree structure. If content portions are as-

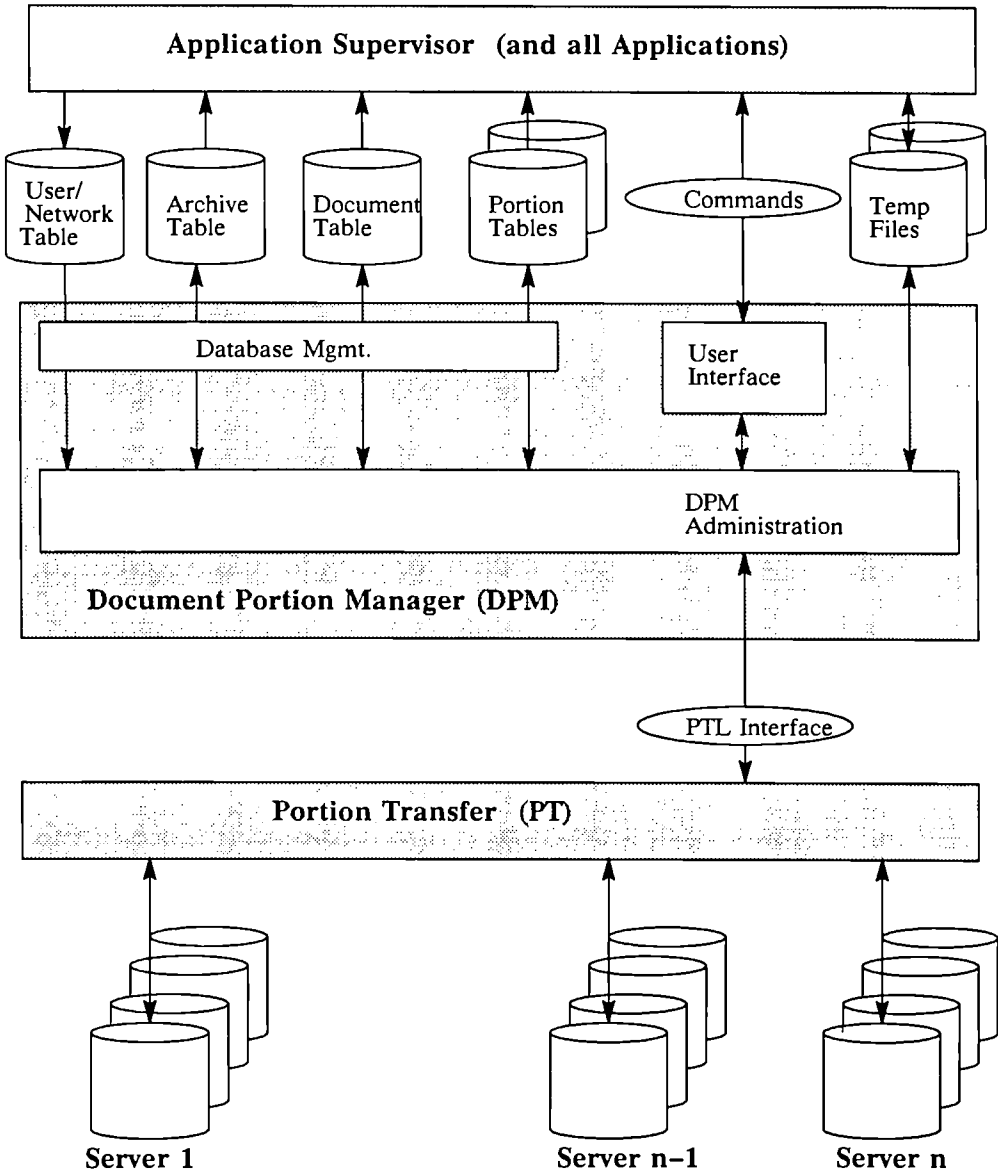


Fig._3: Components and Interfaces of DPM and PT

able by the *Portion Tables*. If a document is retrieved from the servers, the DPM provides the requested portions as temp files to the Application Supervisor. The DPM stores new or updated portions by collecting the content from the temp files.

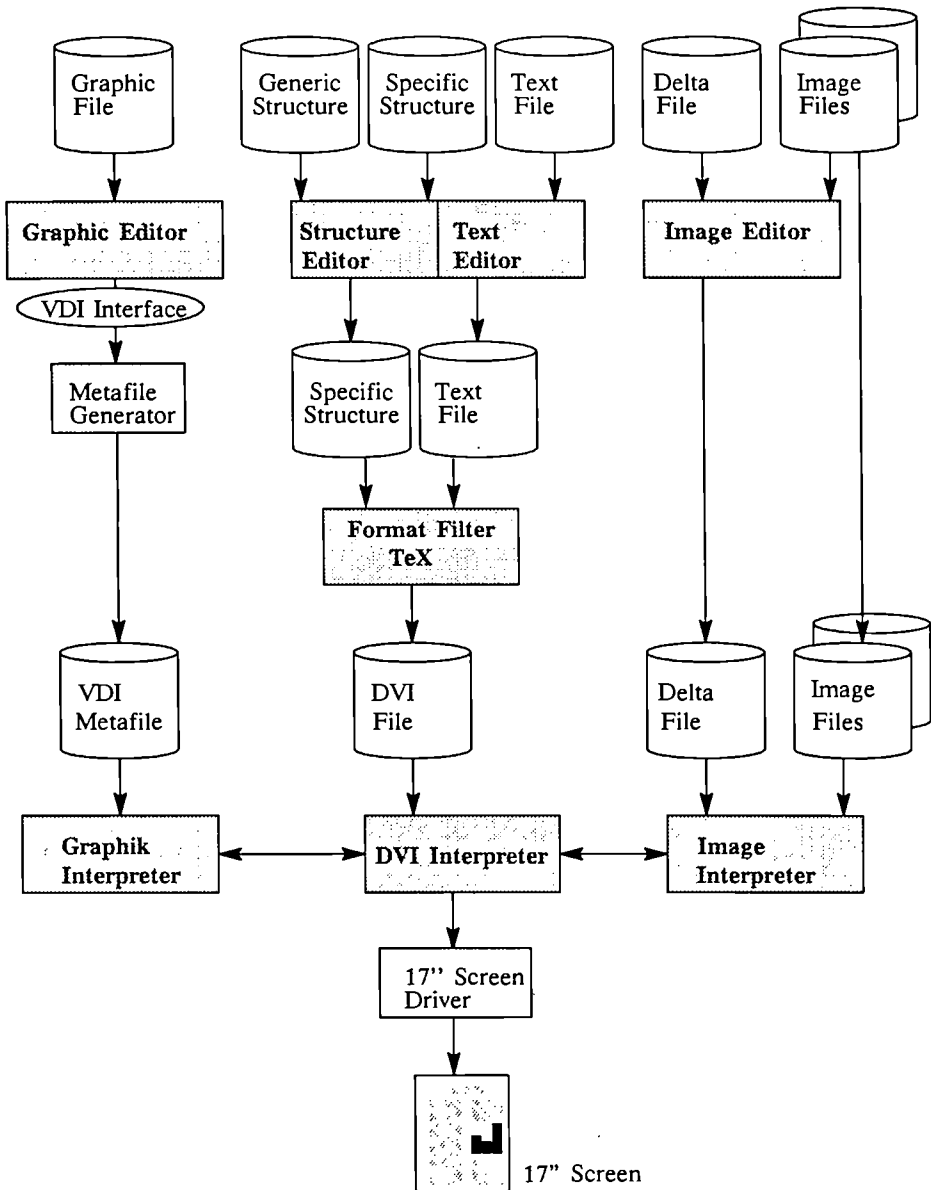


Fig._4: The Document Preparation Process

sociated, the TE will be called to delete them. `Stput()` copies the description of a picked object and assigns it to the actual object, which can be seen as a container for the source object. If content references exist in the source object the TE will be called to insert the text portion(s). `Totxted()` controls the switch to the TE which was explicitly given by the

user (by an entry in the *table of contents* menu). `Togrim()` initiates the evaluation of the content reference for a terminal graphic or image object.

Image Editor

The new image editor supports activities that typically arise in the context of preparing documents: Cutting and pasting images, masking, enlarging, and shrinking. Manipulating images means generating new from old images. So called *delta files* are introduced to store these operations, assuming that:

- * several images have to be generated from some original images, where each original image may be source for several new images;
- * the original images have to be kept for further processing;
- * storing the new images as matrices of pixels requires more space than storing the operations for generating these images;

The image editor supports generating and modifying delta files. That means he manages an *operation tree* which describes the image operations. The concept of delta files gives the following advantages: Keeping the information on how the new image is generated from cuts of source images and further objects enables one to modify or even cancel individual instructions on the delta file. I.e., a delta file represents a structure of the new image which is conceptually similar to that commonly used in graphic systems: In the graphics world the user is able to identify objects very easy (by pointing at them). The situation is totally different with images, the objects (like a pasted image) can in general not be identified on the screen. The introduction of our concept reduces this drawback, as the information is now available in the delta file. The disadvantage is that a new image has to be processed again before each representation.

An *image file* contains a header and a complete representation of an image. The header indicates whether the image is coded by one bit per pixel, using run length (each run length is binary coded and requires two bytes), or using a code as suggested by Takao [3].

Format and View

The TeX software packages had to be adapted to the WS environment. Quite a few problems were related to arithmetic.

TeX writes its formatted texts to *Device Independent Files (DVI-Files)*, which are pagewise transformed by the driver to bitmaps and shown on the 17 inch screen. The DVI Interpreter was extended to cope with multimedia documents. The position and size of a

graphics box and the object name is forwarded to the appropriate interpreter. The graphic or image interpreter inserts the scaled object at the requested place.

4. Hardware

The INCA workstation for advanced office applications must support the generation, manipulation and distribution of mixed mode documents, i.e. documents consisting of image, graphics, text and data information. In particular, with the inclusion of images, the requirements of all components of the system are rapidly increasing (i.e. more bandwidth, processing power and storage capabilities). The existing Nixdorf Workstation PWS-X has been extended to meet the requirements of the ESPRIT project with the addition of a high resolution *pixel display controller and monitor*.

Let us first review the major design issues for the new display controller: screen resolution, frame rate, line frequency, pixel rate, and screen size:

The standard A4 page has a size of 7.8 x 11.2 inches. This leads to 936 x 1344 pixels for a resolution of 120 ppi. The frame buffer is 1024 pixels wide. Taking into account, that standard monitors are built with a format ratio of 3:4, a usable *resolution* of 1024 x 1368 pixel was chosen. In addition, on each side an margin of 48 pixel has to be provided, in order to achieve a white screen margin.

The *frame rate* has to be equal or exceed 70 Hz in order to guarantee a flicker free image on a positive screen. The standard solution of interlacing can not be used, because this would lead to unacceptable flicker, in particular with displays that contain raster images or large size uniformly styled areas. A typical method of reducing flicker is to use long persistence phosphor. This method is producing acceptable results only with a very long persistence, which leads to undesirable smears, if objects are moved on the screen. Our choice is to use the high *refresh rate of 70 Hz in an non interlaced* mode with short persistence phosphor.

The *line frequency* is computed as follows: Starting from our goal of 1368 usable lines we have to add 96 margin lines and 98 lines for vertical retrace, which adds up to 1562 lines. The multiplication with 70 Hz leads to 109.375 KHz. In addition to the 1124 pixel width, we have to allow for 25% horizontal retrace, which leads to a total *maximum frequency* of 157 MHz.

With respect to the *screen size* we have to take ergonomical considerations into account. The height of a capital letter on the screen should be at least 3 mm, and the distance between lines 4.5 mm. The conclusion is, that the screen for a full A4 page should have a

height of 320 mm. To support this request, a display with a 17 inch diagonal tube has to run in *portrait mode*. The main requirements are:

- * display of a complete page A4,
- * resolution of 120 pixel per inch (ppi) on the original document,
- * high frame rate, to achieve a flicker free, positive video display (black characters on white background),
- * hardware support for graphic commands, bit-block operations, windows, and
- * softscroll of the complete display (or parts) in each direction.

120 ppi was chosen to simplify the scaling towards the resolution of printer and scanner devices.

In order to support a flexible window processing it is important, that the hardware does not restrict the number, size, and position of the windows. These requirements can not be fulfilled, if memory planes have to be switched while building up the screen. This would lead to a storage requirement of 130 KBit for each open window, because the window may be of the size of the complete screen. For these reasons the following *choices* were made: Only one memory plane is used, all windows are only under software control, and the circuit has to support simple and fast block movements in the frame buffer.

The display controller meets the following *specifications*:

- * The following command classes of the *Graphics Controller* (HD63484 ACRTC) are used: *draw* circle, arc, line, rectangle, ellipse, elliptic arc, polygon, *fill* random area or rectangle with pattern, *copy* rectangular area inside the frame memory, and *execute* an operation only inside *or* outside a specified area. The normal copying of a 1000 x 1000 pixel wide area, which is not restricted to storage word boundaries, will take one second. But these copy functions are needed for bit block operations, e.g. if the content of a window is moved. The same time of one second would be needed to fill the complete screen with text. For these reasons a special hardware dedicated to bit-block-operations will be introduced.
- * *The Bit Block Operator* component provides fast bit block operations. It required extensive design efforts, and consists of around 6 000 gates.
- * *Frame Memory*: High resolution display controllers impose a high demand upon the bandwidth of the frame buffer. The buffer has to provide time for both updating the display content and the readout to the video monitor simultaneously. This requirement can be met by a new family of dual port video RAMs. The bottleneck for the maximum pixel rate lies now in the video interface, not in the memory access time. We use a chip from NEC, the uPD 41264. The cycle time is 300 nsec for the ACRTC, 280 nsec for the bit block operator in normal mode, and 140 nsec for the page mode. This allows us to move the complete content of the screen in less than

14 ms. The scrolling of the complete screen becomes very smooth and neat, because the buildup of the display is synchronous with the refreshing phase.

- * *High Speed Video Interface*: The video interface picks up the data stream from the shift registers of the RAMs and sends them serialized to the monitor. The interface has to be built in ECL technology due to the high pixel rate of up to 157 MHz. The video signal is purely digital, gray tones and colour are not supported. The brightness is controlled by an additional analog signal. The synchronization signals (one for horizontal, one for vertical) are transmitted in TTL technology.

Beside the VT100 terminal emulation some basic routines for displaying and manipulating image data on the screen have to be provided. Three elementary functions for bit mapping are implemented: The routine `display_map()` copies a rectangular bitmap from the application RAM onto the screen. The parameter mode defines the logical conjunction between the pixels of the bitmap and the destination rectangle on the screen. The function `bitBlit()` copies a rectangular subregion of the screen onto another one. The function `block_fill()` sets, resets, or inverts all bits lying inside the rectangular of the screen.

The *Monitor* requires a high performance on bandwidth of the video amplifier, the line width, focus, and EMC. The specifications for the new controller and monitor are summarized as follows:

display tube:	17 inch diagonal face, 110 deflection angle
frame rate:	70 Hz, non interlaced
usable display size:	1024 x 1368 pixel (without margins)
margins:	48 pixel on each side
display mode:	portrait
line frequency:	109.375 KHz
number of lines:	1464 (incl. margin, excl. retrace)
max. pixel rate:	157 MHz

5. Outlook

The INCA Workstation has been successfully demonstrated in October 1986. This sub-project has now come to a provisional end. In this field of application the new standards have made considerable progress. When the work was started in 1984 two standards had been under development but they both were uncomplete at that time:

- * ECMA 101, Office Document Architecture [4]
- * ISO 8879, SGML [5]

Because ODA was far from being in the stage to be implementable, we took the decision to take the idea of ECMA 101 and implemented it on the basis of SGML. The situation has

changed now, ODA is nearly agreed as an ISO standard, the second draft [6] is send for voting and it is fairly sure that it will be formally accepted early second quarter next year. Additionally this standard will also be included in the new recommendations of Study Group VIII in CCITT first quarter next year. SGML - ISO 8879 - has already the status of a standard. Although in some kind the two standards are overlapping in the field of application and competing with each other, both have their merrits:

- * ODA, in the office scenario, with well define semantic, but with restricted functionality, and
- * SGML, in the area of publishing, with great flexibility but with a lack of semantic for the elements (objects).

At the other hand the office and publishing scenarios are merging. Todays desktop publishing systems are the first step in that direction.

Therefore we decided to continue to work on an editor/formatter which can deal with ODA documents as well as with SGML documents. Additionally some functional enhancements of ODA are needed which we include in our future work in INCA:

- * Distributed document, parts of the document are distributed over the local area network
- * Integration of actual processed data from the database
- * Automatic generation of letters using generic text and actual data
- * Annotation, for electronical reviewing of documents

Some of the results of the first work, which ended last october, have been included and further developed during this year within the office product line of Nixdorf. Especially the high resolution display received a very good response from the market. It was presented at the CeBIT fair in Hannover 1987 and will be shipped to the market beginning next year.

6. Bibliography

- [1] Godbersen, H.P.; Malpeli, F.:
First Results on the Integration of Communication and Application Services in the Office.
ESPRIT '85 Status Report, Part 2, North Holland Publ. Co., p.1145-1155
- [2] Godbersen, H.P.; Koehler, G.; Lewin, Ch.; Seib, J.; Zschoche, G.:
Bearbeitung, Verwaltung und Versand von Dokumenten.
Proc. GI Jahrestagung 1986
- [3] Takao, Y:
An Approach to Image Editing and Filing.
(in:) Picture Engineering (eds.:) King-sun Fu, T.L. Kunii
Springer Verlag Berlin-Heidelberg-New York
- [4] ECMA-101
Office Document Architecture.
Sept. 1985
- [5] ISO 8879
Information Processing - Text and Office Systems- Standard Generalized Markup Language (SGML).
- [6] ISO/DIS 8613
Information processing - Text and Office Systems - Office Document Architecture (ODA) and Interchange Format.
- [7] Kirstein, P.T.:
Planning for the INCA Demonstrator.
ESPRIT '86 Status Report, North Holland Publ. Co.

INCA DIRECTORY SERVICES

S.E. Kille

Department of Computer Science
University College London
Gower Street
London, WC1E 6BT
UK

The INCA project is performing theoretical and practical studies on the use of standard Directory Services to support a wide spectrum of Communication Activities. Two broad classes of service are considered:

1. Directory Service to support management, searching, and complex matching.
2. High performance "Name Service" to support OSI processes.

The INCA specifications show how these types of activity interrelate, and how the standard protocols are appropriate for both types of service. The paper starts by considering the background and goals of this work. A discussion of the relationship to the THORN project is given ¹¹. The paper then considers issues relating to the implementation of systems to provide these services, and in particular problems relating to the provision of highly distributed systems. Finally, the plans for INCA pilots to investigate these areas are discussed.

1. INTRODUCTION

The INCA (Integrated Network Communication Architecture) project is an Esprit research project comprising GEC (prime contractor), ATM, Nixdorf, Olivetti, and University College London (UCL). UCL is a subcontractor to GEC. It is investigating a wide range of areas, including support of MultiMedia workstations, and provision of a range of networking technologies such as the Olivetti broadband net. This paper is not an appropriate place to discuss details of the whole project. The Directory Service activities are a small component of this project, with the bulk of the work being done at UCL. The INCA project is strongly aligned to standards, and so "Directory Service" implies the emerging CCITT/ISO standard, unless there is a very good reason to use something different ⁴. INCA chooses to utilise the standard protocols for all aspects of the work item. This paper does not contain a tutorial material on Directory Services, but assumes some minimum knowledge of the standards. For those not familiar with the general concepts, the paper by Manros is a useful introduction ⁸. The INCA activity on Directory Services is not large (e.g. in comparison to the THORN project), and so only selected areas of the potentially very large problem are tackled. The approach taken to limiting the problem is discussed later.

There are two distinct aspects of this work:

- (1) A study of some of the interesting problems of providing a directory service. This study is to be made by design and implementation of an experimental directory service. The experimental implementation is seen as a demonstration of the viability of the ideas proposed.
- (2) A study of the problems of directory service usage. This is done in the context of all of the other various INCA applications. The INCA Naming Architecture (INCA deliverable 6.1) specified the naming structures to be used by the various OSI services and Applications ⁶. Because it was phrased entirely in terms of the Directory Service Information Framework, this specification was also a service definition for usage of the Directory Service. The viability of this architecture and of the INCA Directory Service will be shown in the INCA Demonstrators.

Directory Services can, in principle, be used to provide a very wide range of services. INCA attempts to show how far this can be reasonably taken. This "Directory Service Zealotry" is a component of both Directory Service design and usage. The naming architecture has attempted to use the Directory Service Information Framework for all of its application level specifications, and implies a wide range of Directory Service usage. It

may turn out, that the directory service is inappropriate for some of this functionality. However, we take an optimistic view as to how far directory service usage can be taken, rather than prematurely ruling out applications. Similarly, the Directory Service will make extensive use of itself to control its own configuration and operation. There should be an absolute minimum of external control information. It is hoped that this may lead to some considerations as to the practical limits of Directory Service usage.

It is useful to understand the relationship of this work to the THORN project ^{11,0}. Three of the five INCA Organisations are also involved in the THORN project (GEC, Olivetti, and UCL). The THORN project has two major aims:

- (1) To implement a Directory Service system conforming to the emerging international standards. Whilst this system is pre-competitive, it is explicitly conceived as the basis for products.
- (2) To gain experience with operation of a Large Scale Pilot Exercise in the communities of some of the THORN partners.

Thus, the concepts of *service* and *product* are fundamental to THORN, whereas the INCA work is primarily experimental, and so can avoid many of the implementation problems of the THORN system. The simplifications taken by INCA are discussed later. The major output of INCA, will be experience in the problems of Directory Services. However, there is liaison co-ordination between the two projects, and so there should be useful cross-fertilisation.

2. THE INCA NAMING ARCHITECTURE

The INCA Naming Architecture had four goals:

- (1) To define the layering used by naming within INCA.
- (2) To define the syntaxes used at the various layers.
- (3) To define recommended name forms for the various INCA applications.
- (4) To define the various mappings which must be provided for the INCA applications; This effectively provides a Directory Service service specification.

Let us consider these targets and their solution in a little more detail. It is important to appreciate that naming is layered in a manner analogous to protocol layering. A layering choice is fundamental to any naming architecture. The solution chosen for INCA is essentially the one being developed for the OSI reference model ². This places requirements at the Network Level and at the Application level. For some INCA applications, it has been chosen to have naming layers within the application layer. The network layer syntax follows the relevant ISO standard ¹. The application layer syntax follows the directory standards.

The recommended name forms for INCA applications are based on the recommended name forms specified in the Directory Standards. Some of these applications are discussed briefly. For Message Handling Services, the focus was on support of user friendly naming, and of routing. The former is fundamental to MHS use of directory services. The latter was chosen because it was not being tackled by the standards bodies, as opposed to User Agent capabilities, Distribution Lists, and Authentication. The work on routing has been described in a separate paper, and so is not repeated here ⁷.

There is also a specification of support for Filestores and Multimedia Documents. It is important to distinguish between the service (access to a filestore containing documents), and a specific protocol being used to access the filestore. Thus there is an application level mapping of "generic filestore" -> set of "protocol specific filestore". The identification of "protocol specific filestore" is the Application Entity Title of the filestore concerned, and so can be used to identify its OSI location. Note that INCA considers Application Entity Titles to be encoded as Directory Names. This is considered to be aligned with the standards, even though it disagrees in detail. The identification of documents within the directory immediately shows that there is an overlap with Information Retrieval (IR) functionality. A Directory Information Tree hierarchy could be constructed according to subject area, with "keys" stored as attributes of leaf entries. This would allow for subject oriented searching by key. Whilst the main function of the INCA use of Directory Services is the ability to locate an explicitly identified document, it may also be interesting to determine how much IR functionality could be provided by use of the Directory Service protocols coupled with an appropriate naming architecture.

The final application of Directory Services considered by INCA is that of network level routing. Whilst this is, traditionally, an application considered to be beyond the scope of Application Level Directory Services, the studies suggest that it is a least possible. It is not clear whether the performance could be brought to an acceptable level. A key component of the approach is to assign, in the manner defined by ECMA, network addresses which will default to correct routing without use of external information ³. This will ensure that any call, and in particular calls between Directory Components, can always be routed, thus preventing deadlocks. The Directory Service can then be used to provide a mechanism to utilise more complex connectivity. Intelligent use of caching would ensure overall performance gain. Network addresses are mapped onto Directory Names in a complex, but general manner, which gives a useful hierarchy to Network Addresses to enable them to be stored as *keys* in the Directory. The lack of *explicit* hierarchy in Network Addresses is problematical, but this

will cause equal difficulties for any general network routing scheme. Routing is accomplished by using full or partial lookup of a Network Address in order to determine an appropriate next hop. These comments are clearly speculative, however it is hoped to perform experiments in this area. The results may be useful in designing a Directory Service tailored to this critical problem.

3. NAMESERVICE AND DIRECTORY SERVICE

The INCA project distinguishes between Directory Service and Name Service. There is no fundamental distinction between them. However, there is a definite qualitative distinction. Directory Service covers a wide spectrum of functionality. There are a number of characteristics which are used to characterise the definition of nameservice. The edges are left fuzzy, because there is no natural cutoff point.

- Nameservice is used primarily within the OSI environment. It is not usually used to handle information external to the OSI environment (e.g. postal address).
- Keys are generally known exactly. There is no requirement for "fuzzy matching", searching, or user oriented "clues" in return parameters.
- Typical queries go from a single key to a small number of values (typically one). There is no requirement to support enormous lists of data as value information.
- Nameservice must be "fast": that is return in a time appropriate to an OSI process. This might be in the range 10ms - 10s, depending on whether the query is local or remote.

The archetypal nameserver query is to map from an Application Entity Title onto a PSAP address. Services which provide mappings analogous to this are already provided by tailored protocols, such as the US DARPA Domain Scheme, and the UK NRS Lookup Protocol⁹. These systems were studied at UCL, to examine requirements and implementation techniques for Nameservice. Such a query would usually be made by an Application Entity prior to the establishment of an OSI association. It can be seen that this is aligned with each of the characteristics.

The observant will note that these characteristics apply primarily to OSI system usage of the Nameservice, which is usually read only. The management characteristics of the same information is much closer to that of the general user oriented Directory Service. The INCA approach is to use the standard Directory Service protocols for both Nameservice and Directory Service. There does not seem to be anything inherent in these protocols which makes them unsuitable for Nameservice. The major problem seen, is making a system which gives all of the searching and control facilities required for management, along with the high speed key to value mapping required for basic nameservice. The INCA Nameservice is readonly, and does not provide searching facilities. Therefore, a very simple database can be used to provide the critical nameservice functionality. The database for the INCA Nameservice is in fact managed by the INCA Directory Service. Thus, there is a close symbiosis between the two components. The Nameservice can be viewed simply as an optimised access mechanism to the INCA Directory Service.

4. SYSTEM DESIGN

The most important thing to note is not what is being included, but what is being left out. Because of the environment, there are a number of things which can be optimised, to allow for relatively rapid implementation. The following are intentionally simplified or omitted:

- Database access. One of the hardest problems with a DSA (Directory Service Agent), is to design how the data is stored on the disk. This is being bypassed by INCA.
- Whilst the system will need to cope with realistic quantities of data, it will not be used with very large volumes of data (viz: thousands of entries, but not millions of entries).
- Some aspects of robustness can be relaxed. Whilst it is important that the Directory retains consistency, it is acceptable for some updates to be lost (in the event of a machine crash or similar). This will simplify the requirements on logging and rollback.
- Performance, particularly in terms of resource utilisation, does not have to be optimised.
- There will be no authentication.
- A simple approach will be taken to access control
- A simple approach will be taken to Schemas

It is not possible to describe all the design, but an overview of the major points is given. The directory service will be provided as a "C" program interface, which can be used by the applications concerned. In the DUA, this will map fairly directly onto the associated protocols. It is intended that the INCA DUAs will not perform DSA referrals (the process where a DUA (Directory User Agent) is referred from one DSA to another). In general, queries will be iterated by the DSA contacted by the DUA. There are two advantages to this:

- (1) Minimisation of the directory function which needs to be embedded in the application. This allows the DUA to be stable and lightweight.
- (2) Allowing the "local" DSA to act as a cache for all local DUAs.

A full implementation of all of the protocols will be provided. This will be achieved by use of an ASN.1 compiler. This should make the protocol aspects, and thus the entire DUA, fairly mechanical. The ASN.1 compiler to be used is the one associated with the ISO Development Environment (ISODE) from Northrop Research and Technology Center ¹⁰.

The master of the Directory database will be stored as an ASN.1 encoded set of files. These files are used for both the Directory Service and Nameservice. The Directory Service DSA is a static process. As an initialisation procedure, it reads in the ASN.1 files, and builds an in core tree of its own part of the Directory. This structure is reasonable, given the constraints of INCA. As the volumes of data are not too large, and computer memory is becoming increasingly cheap, this approach is straightforward and flexible. It is not optimised for any particular lookup, but will perform almost any form of lookup without too much overhead. The problem with most disk structures (at least on conventional disks) is that they are very good at what they are optimised for, and very bad for anything else. This is particularly true for some forms of searching. The proposed memory oriented approach will allow us to see what type of access *need* to be optimised, and so can provide useful input for a later product.

For the Nameservice, there will be derived from the ASN.1 files a hash indexed key -> value mapping database, with the key as the Distinguished Name of an entry and the value as the contents of the entry. This will allow for a Nameservice DSA, which may be either a static or dynamic process, to perform rapid name lookups. A number of optimisations are provided. There will be a second hash file, containing cached information, marked with the time they were entered into the cache. This will be updated by DSAs, to improve performance. The Nameservice will contain only information with global read permission. As there is no modification through the Nameservice, write access is not an issue. Given this, for cases of co-resident DUA and DSA, the DUA will be able to directly access the DSA directory and cache files. This will allow highly optimised access to local and cached data. For this reason, most INCA hosts will choose to run a Nameservice DSA, even if this DSA only holds cached data.

An important feature of INCA is experimentation with distribution and replication. The standardisation work in this area is, in the Author's opinion, unimplementable. Therefore INCA is defining its own approaches to these problems. A consequence of this is that whilst the Directory Access Protocol follows the standard, the Directory System Protocol (the protocol between DSAs, has INCA specific modifications). As a matter of principle, all information to control navigation and replication is contained within the Directory itself. Because of this, it is essential that information in the Directory is replicated, even though this is beyond the scope of the current standardisation activity. A key decision is that all entries have a single master copy, and that all entries below a given entry in the Directory Tree have their master copy at one DSA. This set of information is known as an Entry Data Block. These Entry Data Blocks are replicated. The entry above will contain pointers to master and slave copy DSAs. Each DSA will have an entry which gives a list of the Entry Data Blocks for which it is master or slave. There is a "root" Entry Data Block. DSAs which do not have access to this need to have a pointer to the PSAP address of a DSA which will take them (possibly indirectly) to a DSA which has such access. Otherwise, a DSA needs only to know its own name. This leads to a fairly convoluted bootstrap for "non-root" DSAs, but this is only at initial startup because of the possibility of using cached information.

Updating will be done in two ways. Initially, entire Entry Data Blocks will be copied regularly from master to slave by use of a new remote operation. An optimisation of the operation will only transfer data when there have been no changes. Clearly this will cause problems as the volumes of data grow and will be too restrictive, even within the INCA rules. Therefore, an incremental update will be developed. This will utilise X.400 (P1) to provide reliable multicast transfer of incremental updates between DSAs. Because of the nature of the update operations, the master can assume that they will succeed without the need for a confirmation. If an update error occurs, there has been a serious configuration glitch, which will need external intervention. The use of P1 (as opposed to Reliable Transfer Service) is attractive because of the multi-cast available, in parallel with atomic submission. Thus, a DSA with all its data in memory, can send off one of these operations to update all of the other copies, including its own disk copy.

5. CONCLUSIONS

An overview of the design of the INCA Nameservice and Directory Service has been given. It has been suggested how the standard protocols can be supported, in order to give both the full power of the Directory Service, and optimised lookup for Nameservice. There is a consideration of how a lightweight prototype Directory Service can be implemented, and how non-trivial distributed operations will be performed. This service will be used experimentally at UCL, and as part of a number of INCA demonstrators, which should include each of the INCA applications described: Mail; File Transfer; Document Access; Network Management; Network Routing. It is hoped to be able to present results on this during 1988.

REFERENCES

1. ISO TC 97/SC 6, "Information Processing Systems - Data Communications - Addendum to Network Service Definition Covering Network Layer Addressing," ISO draft proposal 8348/DAD2 (October 1984).
2. ISO TC 97/SC 21, "Addendum to ISO 7498 on Naming and Addressing," ISO Draft SC 21 N 944 (February 1986).
3. ECMA TC32, "Domain Specific Part of Network Layer Addresses," Final draft ECMA standard (February 1986).
4. CCITT/ISO joint working group on Directory Services, "Convergence Documents 1-7 (Tokyo)," Directory Systems (June 1987).
5. S.E. Kille, "THORN (The Obviously Required Nameserver)," IES News, (No 5.) pp. 11-14 Esprit, (August 1986).
6. S.E. Kille, "The INCA Naming Architecture," INCA Deliverable 6.1, Esprit project INCA for the CEC (August 1986).
7. S.E. Kille, "MHS Use of Directory Service For Routing," IFIP 6.5 Conference on Message Handling, Munich, North Holland (April 1987).
8. C. Manros, "Directory: the vital support," Electronic Message Systems - ONLINE Conference, ONLINE, New York (October 1986).
9. P.V. Mockapetris, "A Domain Nameserver Scheme," IFIP WG 6.5 Conference, Nottingham, (May 1984).
10. M.T. Rose and D.E. Cass, "The ISO Development Environment at NRTC: User's Manual," Northrop Research Report (March 1987).
11. F. Sirovich and others, "THORN (The Obviously Required Nameserver)," in To be presented at Esprit Technical Week, North Holland (September 1987).

Project No. 169

THE LION PROJECT: A STATUS REPORT

Angelo LUVISON (*), Gérald ROULLET (**), and Flemming TOFT (***)

(*) CSELT, Via G. Reiss Romoli, 274, I-10148 Torino, ITALY

(**) TITN-CGE, Z.I. La Vigne aux Loups, Rue Denis Papin,
F-91380, Chilly-Mazarin, FRANCE

(***) NKT, Brøndbyvestervej, 95, DK-2605 Brøndby, DENMARK

The paper summarizes objectives and results so far achieved for the Esprit Project 169 aimed at the conception and prototype development of a local integrated optical network (LION) with a transmission rate of 140 and, in a second stage, 565 Mbit/s. The basic system and network concepts of LION embody advanced features ranging from high capacity and throughput to large number of users, from low cost per access port to high reliability, from service integration to interoperability with other local and geographic networks. A demonstration of a LION node prototype is given at this conference, while a laboratory prototype network will be available by the end of 1988 with applications entailing voice, data, video and graphics.

1. BACKGROUND TO THE PROJECT

The "Local Integrated Optical Network (LION)" is a five-year Esprit project, which started in September 1983 with the provisional acronym of LWCS, meaning local wideband communication system. The original consortium was formed by TITN (prime contractor) and CSELT (partner), with the participation of universities from France and Italy, as subcontractors.

The subject of LION is a wideband local network capable of integrating all the communication services of a large enterprise. The network, based on optical fibre technology, will provide an advanced support to office communications. It can also adapt to different environments, e.g. high-tech or intelligent buildings, automated factories, industrial research laboratories, universities and hospitals. The system key features were identified from the outset:

- use of optical fibres with 140 or 565 Mbit/s data rate,
- integration of voice, data, graphics and compressed video ,
- connection of several hundred access points with no a priori constraint on the area size,
- identification of functions suitable for VLSI integration,
- interworking with other private or public networks.

The one-year start-up Pilot Phase pointed out a traffic scenario for telecommunication services and provided the functional requirements. Moreover, the overall system architecture was defined by considering the use of commercially available technologies. Various design alternatives were proposed and carefully assessed with analysis and simulation tools.

The First Phase, with the new acronym LION for the project, saw a broader international participation. NKT from Denmark joined the consortium as a

partner and the University of Patras (Greece) as a new subcontractor of TITN. The main work tackled design problems starting from the identification of specific subsystems. The detailed definition and formal specification of each subsystem was the major activity in this phase. By late 1985, specification and design of the basic hardware modules were completed. In parallel, the analysis of software requirements and the set up of development tools gained ground. The development and testing of the hardware modules were also undertaken.

TABLE I
LION PROJECT CONSORTIUM

COMPANY	ROLE
CSELT (Centro Studi e Laboratori Telecomunicazioni SpA) STET Group, Torino, Italy	Prime Contractor
NKT Elektronik (Nordiske Kabel- og Traadfabriker) Brøndby, Denmark	Partner
TITN (Traitement de l'Information Techniques Nouvelles) CGE Group, Chilly-Mazarin, France	Partner
BTRL (British Telecom Research Laboratories) British Telecom plc, Ipswich, United Kingdom	CSELT's subcontractor
Telefónica Departamento de Investigacion y Desarrollo, Madrid, Spain	CSELT's subcontractor
Polytechnic of Milano Dept. of Electronics, Milano, Italy	CSELT's subcontractor
Polytechnic of Torino Dept. of Electronics and Dept. of Informatics and Automation, Torino, Italy	CSELT's subcontractor
University of Patras Lab. of Electromagnetics, Patras, Greece	TITN's subcontractor
University Paul Sabatier Lab. LSI, Toulouse, France	TITN's subcontractor
University of Paris VI Lab. MASI, Paris, France	TITN's subcontractor

The Second Phase--the current one--started in April 1986 and covers the remaining years up to the end of the project in December 1988. The present Project Consortium, shown in Table I includes now three partners--CSELT (prime contractor), TITN (partner), NKT (partner)--and several subcontractors--British Telecom (Great Britain), Telefónica (Spain), the Polytechnics of Milano (Italy) and Torino (Italy), the Universities of Toulouse (France), Paris VI (France) and Patras (Greece).

The main objective to be fulfilled during this phase is the development of LION prototypes, of growing complexity and performance, to prove LION operation, along with a set of selected but meaningful applications and services. Work packages and tasks of the participants have been clearly defined to achieve a better coordination of available resources towards the prototyping objectives. Moreover, the new subcontractors are in charge of pieces of work specifically tailored to their experience and oriented to broaden the range of LION applications. These objectives imply not only study and design work but also the development of both software programmes and hardware devices. The overall plan of the project, briefly described above, is summarized in Fig.1, where the major events and milestones are also shown.

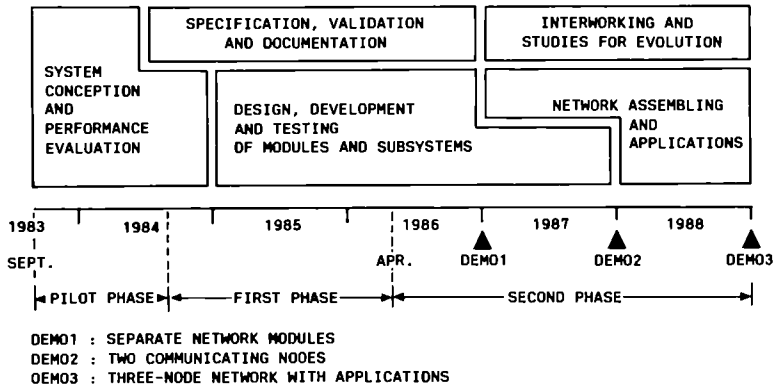


Figure 1
ESPRIT Project LION: overall plan and time phases

2. PROJECT OVERVIEW

The basic goal of LION [1]-[3] is to provide a flexible access to a large number of independent users with the requirement of distributed processing and intelligence. A local area--like an office building, a warehouse, or a college campus, etc.--is considered and the system is intended to sustain several hundred network nodes. In accordance with to the reference architecture shown in Fig.2, each node comprises a network communication unit (NCU) and a medium attachment unit (MAU) connected to the optical fibre. The NCU and MAU implementation is based on hardware and software modules that perform the physical, data link, network and transport functions in conformity with the reference model for Open Systems Interconnection (OSI) of the International Standards Organization (ISO).

The network management architecture has been conceived by setting two goals: 1) careful assessment of recent work carried out by international standardization bodies (ISO, IEEE and ECMA) and 2) definition of a management archi-

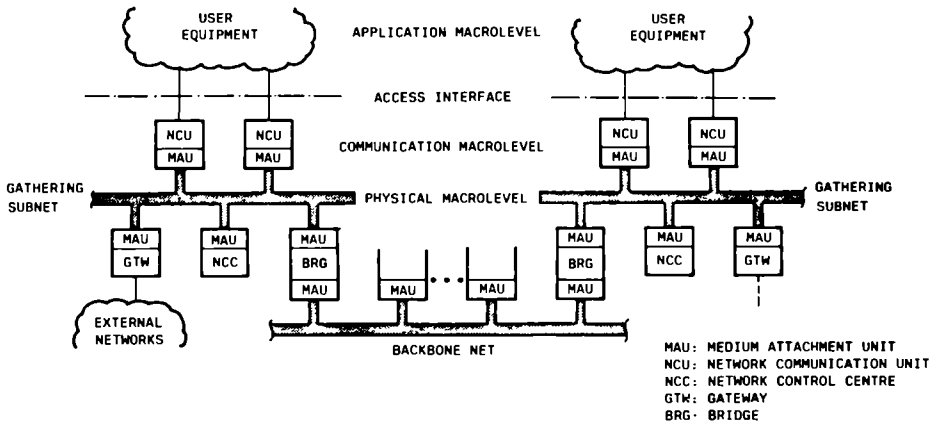


Figure 2
LION reference architecture

architecture consistent with LION objectives (e.g. traffic integration and reliability) and solutions. The main management entity is the network control centre (NCC), which coordinates the network activity with functions of monitoring, diagnosis, network initialization and recovery. In addition, a measurement centre (MC) cooperates with the NCC for traffic pattern generation, performance measurement, statistical analysis and reporting. Local management functions are distributed to each NCU in a specific management module.

The interworking of LION with external networks, e.g. other local-area or long-distance networks, is allowed through suitable gateways. In particular, the gateways to the integrated-services digital network (ISDN), to the TELECOM 1 satellite and to the ETHERNET world have been designed and are being developed.

The main LION objective, i.e. to carry out on one network any type of information an enterprise could generate, engenders severe constraints to the designer. In particular, LION ought to handle a wide range of services in digital form. Therefore, flexibility in service integration is a primary requirement and system efficiency is a need. These factors and the global reliability goal affect the basic choices at various levels in the network architecture, viz. the physical topology, multiple-access technique and communication protocols.

2.1. The Multiple-Access Technique

An emerging solution, known as "hybrid-switching", to integrate services is the provision of both packet- and circuit-switching capabilities on a single system. A method for managing hybrid frames is the synchronous time division multiplexing (TDM), upon which recent LAN proposals, e.g. Fasnet [4] and FDDI-II [5], are based. Both systems provide a fixed channel bit-rate arrangement, usually with a 64 kbit/s PCM frame. To meet the flexibility requirement, i.e. to remove any constraint in the channel size, an asynchronous variable bit-rate technique has been originally conceived for LION. In short, the basic features of the LION hybrid access protocol are:

- Integration of stream-type and bursty-type traffic.
- Provision of a suitable grade of service to each traffic type.

- Real-time traffic handling capability.
- Provision of variable-capacity connections; the capacity can also be varied during a call evolution.
- Different connection modes are allowed: from point-to-point to selective broadcast and conference.
- Any given traffic mix can be operated efficiently, even with heavy load conditions.
- Unused slots are dynamically suppressed. Spared capacity is made available to the incoming users from frame to frame.

References [6],[7] give a description of the multiple-access technique from basic operations to technical solutions.

2.2. Physical Topology

Since the large offered traffic requires the use of optical fibres, the choice for a proper topology is partially conditioned by the multiple-access technique. The hybrid protocol can be adapted to either a bus or a ring topology without relevant changes in the design and operation efficiency. The bus solution has been chosen mainly for reliability reasons. In fact, the ring needs a specific information stripping function to avoid information circulating indefinitely, whereas the bus is physically terminated at both ends. Any error in performing the stripping task results in a total out of service for the ring; on the bus, any signal alteration affects the current frame only.

In a fibreoptic bus topology, either the passive- or active-tap realization is possible for the attachment of stations to the fibre. For a ring only the second solution would be feasible. The former solution shows inherent reliability but is not easily applicable to large networks, because of the high attenuation presented by each optical tap. Therefore, the active solution in which each node acts as a signal repeater, has been chosen for LION.

In summary, two single-fibre cables make a folded-bus topology to connect the sequence of transmitters and receivers. One of the two optical cables--the "read" channel"-- conveys to all nodes the information transmitted on the other cable--the "write" channel.

A specific technique has been developed to overcome the risk of network faults that could be caused by stations or links going out of service, e.g. for a power interruption or a cable break. The technique exploits the configurations the active topology can take and offers full protection against a single fault. If such an event should occur, the network reconfigures itself thereby isolating the failed element. This process takes place automatically under the control of a distributed algorithm. A detailed description of the LION topology and reconfiguration process is given in [8].

2.3. Communication Protocols Profile

As the optical transmission subsystem guarantees a negligible bit error rate, simple communication protocols have been used without sacrificing performance and efficiency. In particular, the following choice has been made:

- Type-1 Logical Link Control (LLC 1) for the sublayer 2b of the OSI link layer, according to the IEEE 802.2 standard. The LLC 1 is set above the Medium Access Control (MAC) sublayer 2a of the OSI link layer.

- Internet Protocol (IP) of the ISO DIS 8473 as sublayer 3c and Null Protocol at the sublayer 3a, for the OSI network layer;
- Class 4 Protocol for the OSI transport layer.

Note that the choice above is consistent with the current trends in standards. However, the design of the software packages is modular so that the inclusion is also possible of a different protocol profile, in the future. For example, an X.25 protocol for the network layer and a Class 0 (or Class 2) protocol for the transport layer will be feasible if needed by different requirements in other applications.

3. WORK PROGRESS

During the last year the effort has been oriented towards the following directions: 1) management and organization of the project new phase and 2) technical activity, especially for the development of LION subsystems, modules and interfaces.

Within the consortium, all the activities has been defined in terms of specifically tailored work packages and tasks (see list in Table II). The tasks identify pieces of work that fit coherently with the modular structure of LION. Moreover, the work is allotted according to the specific background, expertise and capabilities of each partner in the consortium.

To achieve the needed coordination with a reduced effort, a set of guidelines has been prepared regarding the communication among working groups, documentation editing and meetings. The use of a common layout and coding for the documents and a standard documentation path, i.e. from working minutes to task reports and deliverable items, have simplified the editing charge. Keeping track of the whole project progress has therefore been improved.

For the technical activity, three major steps, called DEMOs, have been planned, in accordance with the prototype development from one single node with basic functionalities to a pair of connected nodes and, finally to a three-node network operated at 140 Mbit/s with relevant user applications.

3.1. DEMO 1 (December 1986)

DEMO 1 has been equipped by the end of 1986. It included the main building blocks of the NCU with a set of software packages and hardware modules assembled together. In particular, DEMO 1 consisted of the access control manager (ACM), which roughly corresponds to the OSI MAC sublayer, and a simplified management module (MM) connected by a provisional interface board. The hybrid access protocol operation has been shown in the same environment settled for testing. An ad-hoc frame generation module emulates the MAU functions and a software package implements basic packet transmission/reception procedures, running on the target Motorola boards. A resident application software administers the basic management resources and gives the access to the internal parts of the machine by a friendly operator interface. A photograph of the NCU breadboard is shown in Fig.3.

The development and testing of a stand-alone MAU prototype has been carried out at the same stage with the goal of implementing all the individual MAU functional blocks (i.e. transmitter, receiver, coder, decoder, interface with the NCU, etc.). Some ad-hoc hardware modules have also been developed to test the transmission subsystem.

TABLE II
LIST OF WORK PACKAGES AND TASKS

WORK PACKAGE	TASK NAME	TASK RESPONSIBILITY
PROJECT MANAGEMENT AND ORGANIZATION	Project Management and Organization	CSELT
TRANSMISSION SUBSYSTEM	140 Mbit/s Medium Attachment Unit 565 Mbit/s Low Levels VLSI Design and Development	CSELT NKT NKT
NETWORK COMMUNICATION UNIT	Access Control Module Circuit Module Bursty Data Module Management Module Enhanced Telephone Services	CSELT CSELT TITN CSELT TELEFÓNICA
NETWORK CONTROL MODULES	Network Control Centre Measurement Centre	NKT BTRL
GATEWAYS	Gateway to ISDN Gateway to TELECOM 1 Gateway to ETHERNET	TELEFÓNICA TITN TITN and PATRAS UNI.
APPLICATIONS	File Transfer Service Message Handling System COLORIX Application Video Application Module Telephone Application Module Data Application Module	NKT CSELT TITN CSELT TORINO POLY. TORINO POLY.
SYSTEM INTEGRATION AND DEMONSTRATIONS	System Integration and Demonstrations	CSELT
SYSTEM STUDIES AND LION EVOLUTION	Inter-Company Use of LION Topological Evolution towards a MAN Distributed Call Management for Service Integration Bridge Architecture Protocol Performance Evaluation Managing a Multinetwork Communications Systems	BTRL MILANO POLY. MILANO POLY. NKT PARIS UNI. TOULOUSE UNI.



Figure 3
Photograph of prototype modules for DEMO 1 (December 1986)

3.2. DEMO 2 (December 1987)

DEMO 2 will be a demonstration of node-to-node communication using the 140 Mbit/s transmission subsystem. The demonstrator will consist of two nodes equipped with all the modules envisaged for the final configuration. The nodes will be connected through an optical cable in the laboratory environment.

The NCU prepared for DEMO 1 will be integrated with a complete DMA buffer controller board and equipped with interfaces towards the other internal modules. New modules will be added (Fig. 4), viz. the circuit module (CM), which

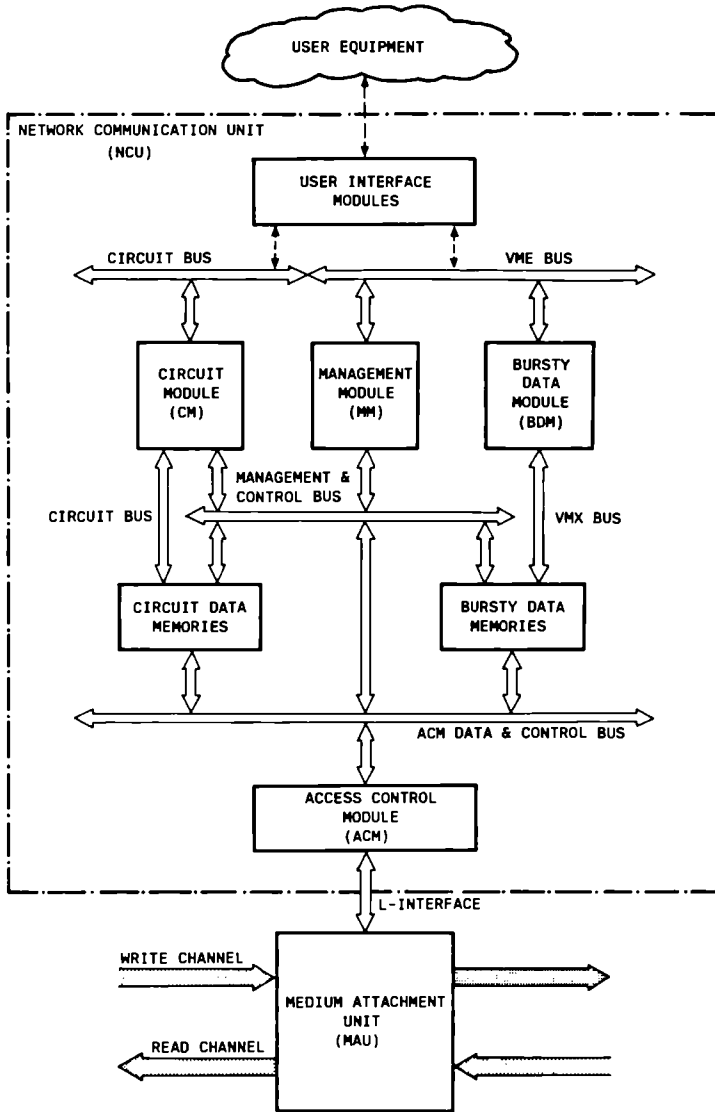


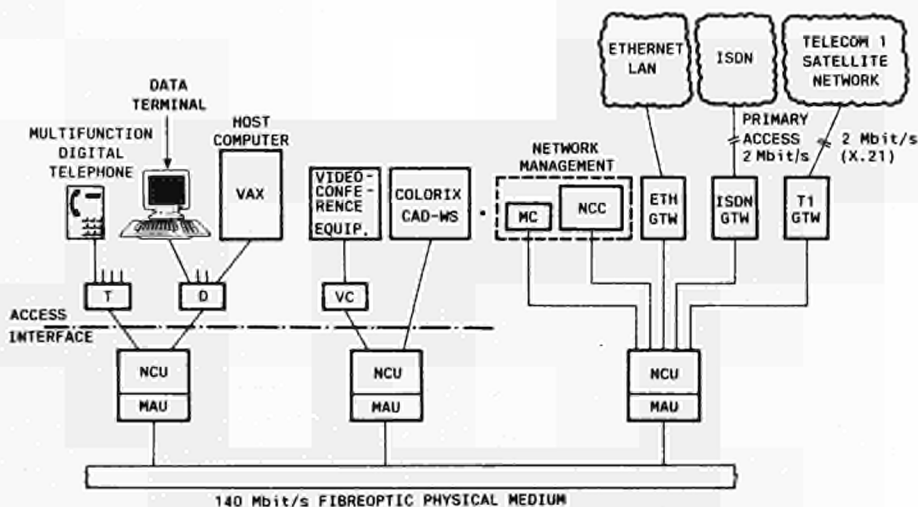
Figure 4
Architecture of a LION node for DEMO 2 (December 1987)

controls the circuit data transfer within the node, and the management module (MM), which performs the basic control functionalities and handles signalling for circuit communications. About the bursty data module (BDM), which processes the data protocols, the demonstration will show the basic functionalities, for example LLC 1 sublayer. However, the same hardware will be used in the final prototype. The MAU in DEMO 2 will show all its capabilities, e.g. those needed to reconfigure the network after a failure [8].

3.3. Final Prototyping Activity and DEMO 3 (December 1988)

DEMO3 will provide a complete network demonstration (Fig.5). Three network nodes will operate under the control of the NCC on a 140 Mbit/s optical cable. Each node will have user interfaces with different features in relation with the mix of traffic supported for the demonstration. Actually, a different throughput is required to handle the NCC, gateways, host computers, data terminals and telephones. Moreover, an additional objective of the final experiment is to point out the NCU versatility and flexibility in adapting to a wide range of services, including file transfer access method (FTAM), message handling system (MHS) and computer-aided design (CAD) connection. The gateways towards external networks, i.e. ISDN, TELECOM 1 satellite and ETHERNET LAN, will also operate on the prototype network.

Two relevant experimental tasks will complement the DEMO 3 work. The first regards the integration of LION physical levels by using application-specific integrated circuits (ASICs) to reduce the chip count and power consumption, in the 140 Mbit/s prototype. The second task covers the design and laboratory development of the transmission subsystem at 565 Mbit/s (the main blocks of the ACM will also be included). ECL devices for the MAU and CMOS technology for the ACM will be used at this higher rate.



T, D, VC: TERMINAL ADAPTERS

Figure 5
Final structure of LION demonstrator DEMO 3 (December 1988)

4. LION AT THE ESPRIT CONFERENCE 1987

The LION project consortium participates to this conference with a demonstration of a node prototype operating with a set of significant applications (Fig.6). The purpose is to show the LION capability to provide the access to different sources, from stream to bursty traffic, from narrow-band to wideband (2 Mbit/s) communications.

This prototype configuration has been carried out to cope with the key objective of the LION project, that is to develop a high-capacity system offering advanced performance as for services, throughput, number of users, reliability and modularity. The development effort has been concentrated on the issues described below.

The fibreoptic transmission subsystem provides a reliable physical layer service. The basic module is the MAU which is placed with the NCU in the same cabinet of the node prototype. The MAU also provides a distributed reconfiguration capability through an internal switching matrix, which performs the connection between the NCU and the optical transmitters and receivers depending on the actual node configuration. In particular, each MAU can act as both the head and folding point of the active bus that supports the LION topology. This makes possible, even with one node, to show meaningful features of the network.

The high performance, required by an advanced local communication system, is mainly provided by the NCU. During the development stage, functional modularity has been pursued to separate access-protocol dependent hardware from upper-layer communication functions, thus allowing a technology free evolution without additional design effort. The lower-layer hardware has been developed with medium- and low-scale integration devices to test fully all the accomplished functions, even though a further integration step is currently under development. Several test and measure equipment are connected to the node cabinet. The LION behaviour in a real operation environment, encompassing both transmission and multiaccess capabilities, can therefore be checked.

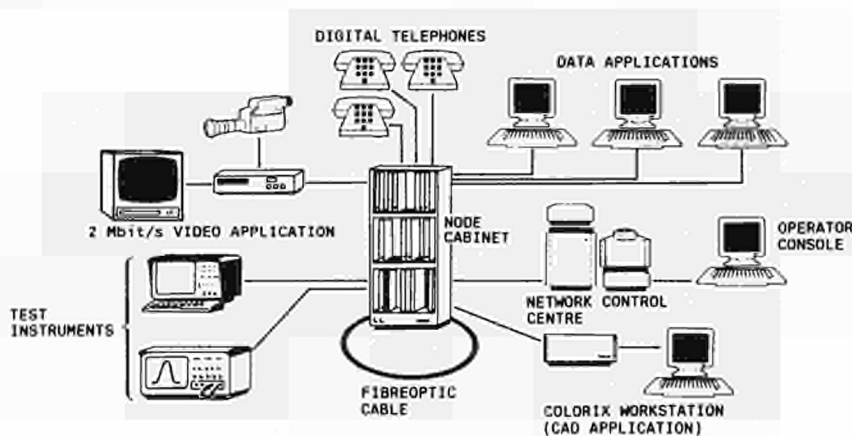


Figure 6
LION and applications presented at the ESPRIT Conference '87
(September 1987)

The remaining modules in the cabinet provide the processing power for management and communication tasks. A multiprocessor system--based on 68010 and 68020 microprocessors--performs the real-time management of local node resources, while assuring a global throughput of about 300 packet/s.

Four special-purpose interface modules are connected to the described system to manage user equipment. The LION node gives a high degree of flexibility about user connection: dedicated cards can be freely used to run different applications, provided compatibility on the system bus is warranted. Firstly, video telephony and video conferencing are offered to wideband users, thus meeting a most desirable service in the future office environment. Video codecs are used to convert the analogue video signal into a 2 Mbit/s compressed digital data flow that feeds the network. The demonstration points to the real-time management of the video service by using a simple TV camera and monitor equipment both connected to a video PAL codec and communicating each other through a LION stream-type connection. However, the future provision of gateways towards external networks (e.g. ISDN) will give the facility of video connections not only locally among LION users, but also between remote sites.

Another promising application of wideband communication systems is in the CAD area, where high-definition video images have to be transferred in near real-time between workstations and mass storage devices. This type of application is well represented by COLORIX workstations connected through LION. A videodisk is used as image source, while from a console an operator can supervise the various operations, e.g. initialization, videodisk control, digitalization of images from the videodisk and image transfer from one COLORIX station to the other.

The provision of the telephone service is indeed a basic capability of the prototype. Moreover, telephone communications will engender a heavy load for the signalling software of the node. An interface card capable of connecting three analogue telephone sets samples the voice input at 64 kbit/s and sends directly the data to the NCU circuit memories. Several of these modules can be simultaneously connected to the node without reducing the node performance. A microprocessor-based version is under development to support enhanced services typical of PABXs.

The access to data applications is currently provided through low-speed serial lines managed by a processor module. It handles connections for terminals and hosts; moreover, it makes available the collection and visualization to users of information on the internal state of the node. A prototype of the NCC based on software modules running on a MicroVAX system is also present with a demonstration of the operator interface and statistics features. The same computer provides host functions for data applications.

5. CONCLUDING REMARKS

In parallel with the experimental objectives, several system studies are being performed with major contributions from the universities and other subcontractors.

A task will supply the requirements and functional specification when LION is used in an inter-company or multitenant environment. The network in the so-called "intelligent buildings" can be shared by small-size companies [9]. This application raises new and more severe constraints mainly in the management of the network (e.g. for controlling common communication resources and services). It also implies specific security requirements to realize a "trust environment", which satisfies at a minimum the privacy of each firm sharing the network. A basic access control system providing capabilities like user

authentication, e.g. with smart-cards, passwords and key management, has been specified. A public key-protocol for mutual identification and the data encryption standard (DES) for secure data exchange have been envisaged. Finally, charging problems and a network database to be shared among the companies are being analyzed.

Another field of application of networks like LION is represented by metropolitan area networks (MANs), where high-speed connections and services are to be offered to heterogeneous distributed users [10]. A MAN user--characterized by a mix of circuit and packet traffic--may be either a simple station clustering a group of terminals or a communication system like a PABX or a LAN. New topologies and call management strategies for service integration are being studied and evaluated in the light of the current research trends in the MAN field.

The advantage of using artificial intelligence methods is also a study topic for the network management. It has been applied to envisage a more refined control of the network in relation with the real-time management of a multi-network environment. In addition, advanced methods for evaluating the performance of communication protocols are under study.

All these activities, to be completed during the last year of the project, are oriented to enhance the scope of LION and to increase its performance and versatility. They should be seen as guidelines for LION evolution towards new applications, raised by the growing demand for advanced communication services.

Commercial exploitation of LION and its results is expected after the project period. Negotiations between the partners will take place in the following year, dealing with organization aspects, i.e. participation in a possible new consortium, work share, timing and test sites. Important technical issues will include the definition of a possible product, technological choices, user interfaces, offered services, performance, etc. Commercial factors, like marketing and price strategy in relation with the selected application environment, will also be assessed. Both engineering and commercial approach will aim at developing LION from a tested laboratory prototype to a product in the communication marketplace.

REFERENCES

- [1] L. Lambarelli, A. Luvison, D. Roffinella and M. Sposini, "Service integration in wideband local area networks: problems and system solutions," in Digital Communications, E. Biglieri and G. Prati, Eds. Amsterdam, The Netherlands: North-Holland, 1986, pp. 315-327.
- [2] A. Luvison, G. Rouillet and F. Toft, "A high-capacity multiservice local area network," in Preprints 3rd Int. Conf. on New Systems and Services in Telecommun., Liège, Belgium, Nov. 12-14, 1986, pp. 2.23-2.30.
- [3] A. Luvison, G. Rouillet and F. Toft, "The ESPRIT Project LION: an integrated multiservice local network," in Proc. 3rd Int. Conference on Data Commun. Systems and Their Performance, Rio de Janeiro, Brazil, June 22-25, 1987, pp. 35-48.
- [4] J. W. Mark and J. O. Limb, "Integrated voice/data services on Fasnet," AT&T Bell Lab. Tech. J., vol.63, pp.307-336, Feb. 1984.
- [5] F. E. Ross, "FDDI--a tutorial," IEEE Commun. Mag., vol.24, pp.10-17, May 1986.
- [6] L. Lambarelli, A. Luvison, D. Roffinella and M. Sposini, "A high-performance multi-service local communication system," in Proc. IEEE Global Telecommun. Conf. (GLOBECOM '85), New Orleans, LA, Dec. 2-5, 1985, pp.457-461.

- [7] G. Audisio, F. Ferrero, L. Grossi and P. Marchisio, "An experimental integrated multiservice LAN," in Proc. Mediterranean Electrotech. Conf. (MELECON '87), Rome, Italy, March 24-26, 1987, pp.275-279.
- [8] A. Brosio, F. Gagliardi, L. Lambarelli, G. Panarotto, D. Roffinella and M. Sposini, "A reconfigurable high-speed optical system for integrated local communications," IEEE J. Select. Areas Commun., vol. SAC-3, Nov. 1985, pp. 825-834.
- [9] T. B. Cross, "What makes a building intelligent?," Data Commun., vol.15, pp.239-255, March 1986.
- [10] O. W. W. Yang and J. W. Mark, "Design issues in metropolitan area networks," in Proc. IEEE Int. Conf. Commun. (ICC '86), Toronto, Canada, June 22-25, 1986, pp.28.6.1-28.6.5.

Object-Oriented Architecture For Distributed Office Systems

C. Horn (1) and S. Krakowiak (2)

(1) Distributed Systems Group, Dept of Computer Science, Trinity College, IRL-Dublin 2
(horn@tcdcs.uucp)

(2) Laboratoire de Genie Informatique, IMAG BP 68, F-38402 Saint Martin d'Herès Cedex
(krakowiak@imag.uucp)

This work has been carried out under ESPRIT contract 834, Construction and Management of Distributed Office Systems. The authors gratefully acknowledge the contributors to the Comandos O-O Working Group: Bull - R. Balter, J. Bernadat, E. Paire, G. Vandôme; IEI - E. Bertino, R. Gagliardi, C. Meghini, F. Rabitti, C. Thanos; INESC - J. Alves-Marques, J.P Cunha, P. Guedes, P. Pinto; LGI - M. Meysembourg, C. Roisin, X. Rousset de Pina, R. Scioville; Olivetti - A. Billocci, A. Converti, M. Farusi, L. Martino, C. Tani; TCD - V. Cahill, A. Donnelly, A. El-Habbash, E. Finn, N. Harris, G. Starovic, B. Tangney, I. White.

1. - Introduction

The overall aim of the Construction and Management of Distributed Office Systems (Comandos) project is to provide an infrastructure for the design, installation and operation of distributed office systems. Comandos started in March 1986, involving seven partners and three subcontractors.

A major and strategic goal of Comandos is to define an architecture for the operational environment of distributed systems for offices and other application areas. This architecture should be implementable in both existing and new system environments, incorporating emerging technologies, and within the framework of existing applications. The architecture emphasises the integration of technology from distributed operating systems, distributed programming languages and distributed databases. The first major definition of this architecture will be delivered to the CEC in September 1987.

The architecture is object-oriented since this greatly assists the extensibility of the system, as well as allowing services and information to be abstractly described. Existing applications can also be captured as (possibly large) objects implementing particular interfaces. The architecture is based on three main features:

- a common and extensible type model, which allows data representation and implementation to be changed as necessary, while maintaining abstract structural and behavioural properties. Thus this model integrates both the database and general programming language environments.
- a computational model which allows distributed programs to be defined. Atomic transactions and exception handling are combined to ensure distributed concurrency control and consistency management, as well as aiding fault tolerance.
- a storage management system which may be extended by new storage and access

This paper describes the major functional units of the Comandos architecture and explains how they combine to provide an extensible infrastructure to support the execution environment for distributed systems for offices and other applications. Section 1 describes the overall goals for the architecture and explains why an object-oriented approach was adopted. Section 2 introduces the overall approach to the architecture using the concept of the Comandos Virtual Machine, which is described in more detail in section 3, and the architecture functional components, which are likewise discussed in section 4. Section 5 gives a brief summary of our implementation plans. Section 6 discusses the relationship between Comandos and certain other projects. Finally we draw some conclusions in Section 7.

2. - An object-oriented architecture

Comandos is targeted for loosely coupled distributed systems of workstations, servers and processor pools, using internets based primarily on LANs. Connection to slower speed WANs will also be supported. There are many research and development teams worldwide building software support for such hardware environments. Operating systems such as Mach-1 [Baron86] and V [Cheriton84] offer relatively low level and efficient distributed message passing support. Distributed multi-language environments such as Saguaro [Andrews87] aim to ease the programming of distributed applications. Distributed operating systems such as Locus [Walker83] have included low-level support for data consistency, and Argus [Liskov83] has extended this by linguistic support for applications operating on long-lived data. Extensible DBMSs such as Exodus [Carey86] and Starburst [Schwarz86] allow application developers to tailor the data management support.

Comandos has borrowed many ideas from these systems and is attempting to integrate them into a unified platform for programming distributed applications which can handle persistent data. An object-oriented approach appears to be beneficial for the reasons stated later below. Other researchers are also taking such an approach, but few appear to have the embracive goal of Comandos. Emerald [Black87] and Distributed Smalltalk [DeCouchant86] provide support for distributed programming in the context of a single language. Gemstone [Maier86] and Trellis/Owl [OBrien86] are integrating data management support into object oriented languages, but again are mono-lingual and currently have limited support for distribution. Galileo [Albano85] provides a powerful data modelling system for persistent data but is not yet distributed.

The goals of Comandos are thus extremely ambitious, but equally would be extremely beneficial and significant if they are achieved. There is currently no standard platform for programming distributed applications which can handle persistent data: however its availability would be highly advantageous.

Object orientation was adopted in Comandos for a number of reasons. Abstraction is not only an important programming concept, but it also aids distributed systems by allowing the implementation and configuration of common services to be hidden from clients. Bundling executable code and data together into a single entity also aids the construction of such systems via the modularity which results. Representing the same entity via different interfaces aids protection mechanisms. Code re-usage and incremental development aids programmer productivity but also allows implementors and administrators to tailor and extend a common system to their own requirements. Object oriented data models provide a rich framework in which to capture complex structures such as the multi-media documents found in office environments. Entity identity removes certain ambiguities. Finally, the approach is appearing in international standardisation activities such as ISO ODA for multi-media documents, the ECMA DASE proposal and, to an extent, the ISO ODP work for distributed environments.

Nevertheless, and despite the origins of object orientation twenty years ago in Simula-67 [Dahl68], in our view object orientation is still a relatively immature technology. While its benefits have been amply demonstrated by environments such as Smalltalk-80 [Goldberg83] and Flavors

and in a distributed system with potentially very many nodes. Some object based systems have relied on special hardware or firmware support for method invocation, but techniques have now been developed to handle invocation efficiently on conventional hardware: it remains to be proven how well these techniques can be developed to handle distributed and persistent objects, in which an object reference can be significantly longer.

Further important considerations for the Comandos execution environment are integration both with existing environments and with management tools. In particular, co-existence with UNIX is crucial and it is essential that UNIX applications can be run in Comandos, although re-linking of object code may be required. Management of a large scale distributed environment can be very complex, and is exacerbated by distributed data management. Our view is that generally it may be beyond the ability of human administrators to control such complex systems accurately, and machine assistance will be required. This problem, and that of providing feedback to Office System designers, is also being studied by the Comandos consortium [Ness87].

The Comandos object oriented architecture work has been divided into two main activities. Firstly the Comandos Virtual Machine describes the environment for application programmers in Comandos systems. It in turn consists of two further chief concepts, the Comandos Type Model and the Comandos Computational Model. It is our intention to provide linguistic support for both of these models, both by building utility libraries for some selected existing programming languages, and also by developing a new language framework which makes it more natural to express programs for the Virtual Machine.

The second task is to describe the architectural components which can together implement the Virtual Machine. These architectural components can initially be functionally viewed, in terms of the interfaces they provide to one another and to the Virtual Machine. Next these components can each in turn be decomposed into a number of internal components, responsible for implementing a specific mechanism in the overall architecture. Finally the interfaces between peer components on different machines can be identified: these command-response interactions are the basis for distributing implementations of the architecture in specific environments. Naturally a minimal implementation level can be identified which every node must provide: a minimally configured node can then gain access to the full complement of functionality available at remote sites.

The Virtual Machine and functional components are now discussed.

3. - Comandos Virtual Machine

3.1 - Type Model

In Comandos, unlike some object-oriented systems, objects are typed. Further, static type checking is exploited as far as is possible, both to assist code generation for conventional hardware architectures and to detect certain programming errors as early as possible.

Each object is normally expected to have state, or *instance data*, and a number of operations, or *methods*, which can be invoked for examining and manipulating the current state of the object. The object has an associated *Type*, which represents the visible properties and behaviour of the object. Each object also has an *implementation* which describes how the abstract interface represented by its type is implemented. A type may have several different implementations, perhaps reflecting different algorithms used to represent the type. There are various primitive types provided directly in the model, together with a number of conventional type constructors for building new types.

In general, it may be possible that instances of one type (regardless of their implementation) may be able to be used *as if* they were instances of another type. This occurs if there is a *subtyping* relationship between the two types. One implementation may also *inherit* representation from one or more other implementations, in which case there will also be a subtyping relationship between

this implementation and the others. Both types and implementations may be parameterised by further types, and so generic types are also supported.

Inheritance, subtyping and genericity all encourage code re-use. However in a multi-user distributed system the extensibility demands may be more rigorous than those found in conventional single-user object oriented environments. Some researchers are considering schemes such as *delegation* [Lieberman81] to support extensibility in distributed environments. Comandos is taking a more conservative approach based on static construction of inheritance hierarchies, and distributing executable copies (possibly in different versions for different hardware instruction sets) of implementations. Nevertheless, totally static checking may be too inflexible, and we are considering concepts such as *enhancement* [Hom87] to augment the scheme.

In Comandos, a *class* is a group of objects of a specified type, or of any subtype of this type. The class and the objects within the class are persistent. A Class object supports insertion and removal of objects from the class, as well as iteration over the objects currently within the class. In addition, a group of classes may have a query predicate defined over them to associatively search for objects satisfying the predicate. Structural as well as content based queries should be supported, since these are particularly useful for multi-media document systems. Classes may be related by integrity constraints so that invariants defined over a group of classes can be maintained by the system. Class objects may also be related by sub-classing relationships, similar to the subtyping relationships of types.

3.2 - Computational Model

In Comandos, there are two fundamental kinds of objects: *active* and *passive*. Independently of any current operation invocations on themselves, active objects may change their own state and invoke other objects: conceptually they have a "process" (at least one) inside of them which appropriately maintains the state of the object. Passive objects can only change their state as a direct consequence of an operation invocation.

Most Comandos objects are in fact passive. A *job* represents the processing (possibly in parallel) of objects and consists of one or more *activities*. An activity is the fundamental active object in the Comandos environment. An activity is created to invoke and execute a specific operation on a specific object. The creating activity continues in parallel with the new activity as soon as the new activity is created, and before the new activity necessarily terminates.

A *context* is a dynamically varying collection of objects located together at the same node: in practice a context is an abstraction of a machine memory address space. A job may have more than one associated context, and likewise an activity within that job can be present in more than one context. This leads to the concept of *diffusion*, in which a job (or rather activities within that job) can subsume additional contexts in additional processing nodes when appropriate to do so. Thus a job is not necessarily confined to a single node.

Objects may be shared in a number of ways, depending on the consistency requirements for concurrent usage. *Atomic objects* can only be examined or modified via *atomic transactions*. *Synchronised objects* maintain their own consistency with respect to concurrent operation invocations, and have functionality similar to a classical Hoare monitor. Comandos does not make any guarantees for the consistency of *non-synchronised objects*, and assumes that they are such that no guarantees are required, or that applications can make their own arrangements to maintain consistency.

Within a single node, several jobs may be present in their own contexts. An object which is shared between two or more of these jobs will appear in each of the contexts concerned. Thus modifications made to this object by one job will be visible to others: however the delay, if any, before these changes actually become visible depends on the synchronisation category of the object.

Activities within jobs invoke operations on objects: thus communication between jobs may be via mutually shared objects. These objects may for example represent a message port [Baron86].

Comandos does provide *channels* as one particular category of objects for communication. A channel is a unidirectional communication path between objects, and is somewhat similar to a UNIX pipe. A prime motivation for channels is to model stream-oriented i/o devices, but they may be used for general communication as well.

Triggers are also supported as a means of automatically initiating activities once associated predicates, defined over objects, become true.

The transaction model for Comandos is intended to support both relatively short duration atomic transactions, and long duration (of the order of days or even weeks) transactions, which support persistency of control (as well as data) via intermediate checkpoints. The combination of long duration transactions and triggers is expected to be a powerful basis for automated *Office Procedures* [Baker87].

4. - Functional Components of the Architecture

4.1. - Persistent objects

An important goal of the project is to provide a unified framework for the management of objects, which supports the viewpoints of both databases and general purpose programming languages. Usually, programming languages deal with volatile objects, i.e. objects whose lifetime is limited to the execution of a program, while database languages deal with persistent ones, i.e. objects whose lifetime is independent from that of the programs which use them. The Comandos architecture aims to support both mechanisms in an uniform way.

The persistence of an object derives from the lifetime of the references to it. More precisely, if the existence is assumed of a permanently accessible "eternal root" (an object that is never deleted), a persistent object is defined as one that is only accessible from the eternal root via class or instance variables, while a volatile object is only accessible from some volatile root (an activity or method variable). Persistence is thus a logical property of an object, which depends neither on its type nor on the storage medium on which the object resides.

A number of solutions have been proposed to deal with persistent objects in programming languages: extensions to programming languages, persistent virtual memory, and object-oriented databases. The persistent virtual memory approach was chosen in Comandos because it is coherent with the global strategy of the project, i.e. to define a new virtual machine and to support it at the operating system level.

One view of the persistent virtual memory (as presented, for instance, in [Thatte 86]), is a single virtual address space, in which every part of every object is always directly addressable. This approach was initially proposed for a centralized system; it was rejected for Comandos, for two main reasons, based on the current state of the art: a) an efficient implementation of a single, universal virtual address space on a heterogeneous distributed system does not seem technically feasible; and b) a conventional virtual address space is insufficient to address all objects in a large distributed system.

At the system level therefore a two-level storage model is being proposed for the support of persistent objects. At the upper level, the *Virtual Object Memory (VOM)* contains the objects that are used by the currently active jobs. This includes persistent and volatile objects; some of the objects may be garbage (i.e. inaccessible). At the lower level, the *Storage SubSystem (SS)* contains all persistent objects. The organization of the VOM and the SS is described below in sections 4.2 and 4.3. Two remarks are however in order:

- the distributed nature of the system is not apparent at this level of description; however, distribution is supported by the components of the architecture, as explained in the next sections:

- at the application level, there is no distinction between the VOM and the SS; objects are moved from the SS to the VOM "on demand" and are written back to the SS either explicitly, or at job or transaction completion, or when the VOM is cleaned to free space.

At the system level, every object is identified by a globally unique, machine independent identifier, which we call its *Low-Level Identifier (LLI)*. These are used to locate objects anywhere in the system. They provide support for object identity and for object migration. They are the basis for dynamic binding of objects, such as clients and servers, together and for supporting exchanges of abstract interfaces. LLIs may also be supplemented by hints to accelerate the locating of objects on both secondary and volatile storage. Finally higher level naming services may be used to map human readable names to LLIs.

4.2. - The Virtual Object Memory (VOM)

The Virtual Object Memory provides addressing environment for all jobs. An object that is mapped in VOM may be accessed by machine (virtual) address. The VOM may be regarded as the collection of all address spaces of all jobs in the system. The VOM maintains a mapping of LLI to virtual address for all objects that are in current use by a job. Thus, when an object is passed as a parameter to a job (e.g. by a name server) using its LLI, the VOM may find out whether the object is already mapped. If not, the object is retrieved from secondary storage. The object is then added to the context of the job, mapped in virtual memory, and the mapping table is updated. If there is no space left in virtual memory to map a new object, some older objects may be stored back into secondary storage in order to retrieve the virtual space which they occupy.

The above description applies to non-atomic objects. The processing of atomic objects also involves the Transaction Manager described in section 4.4.

The VOM is implemented as a collection of virtual address spaces at the various nodes of the system. These address spaces are managed using conventional techniques, such as swapping or paging. The secondary storage used for swapping is distinct from that used by the Storage SubSystem.

4.3. - The Storage SubSystem (SS)

The Storage SubSystem (SS) is responsible for the management of persistent objects. It provides the only means of access to persistent storage. The Storage SubSystem is organized as a set of containers. Containers are mapped on physical secondary storage devices; these mappings may be modified, under the control of system administrators.

Each container is organized as a set of contiguous storage units called segments, which consist of an integral number of disk blocks. Segments are used to store objects. By default, each object is stored in a single distinct segment. However, a number of inter-related objects may be stored into the same segment, so that they be accessed in a single i/o operation when necessary. On the other hand certain objects may be partitioned over a set of segments, particularly so as to aid query processing. Efficient manipulation of objects both in VOM and SS will be crucial to the performance of Comandos implementations.

The LLI of a persistent object is used to identify both its container and, within this, the position of the object. Each container has an LLI map, which is keyed by the LLI value and contains an entry for each object in the container. An entry for an object in the LLI map contains the position and length of the segment which stores that object, together with additional information in the case of clustered and multi-segment objects. The LLI map also identifies an any access method by which a particular group of objects may be retrieved and stored. New access methods may be introduced by system administrators. Finally, various hints may be given to the Storage SubSystem to advise the optimal policy for storing particular objects.

Since secondary storage is a finite resource, an important consideration is the reclaiming of storage for objects that are no longer used. Object-orientation and distribution make this problem

especially difficult. In an object-oriented system, an object cannot be easily deleted since many other objects may hold references to it: garbage collection is used to detect the objects that are no longer reachable from the eternal root. Global garbage collection is extremely difficult in a large scale general purpose distributed system, and notably it is usually infeasible to halt the entire system for global garbage collection.

A direction is suggested by Lieberman and Hewitt's hypothesis [Lieberman 83]: as objects age by surviving garbage collections, they are less likely to be garbage in the future. Although we considered various schemes, including weighted reference counts and mark/sweep algorithms, the proposed mechanism for Comandos is in fact *not* to conduct garbage collection of secondary storage. Instead unused objects are progressively moved to higher levels of container hierarchies, perhaps implemented on slower storage devices. Objects which have not been used for a long period may eventually be deleted. This mechanism is called *ageing*: in the spirit of Lieberman and Hewitt, as objects are used they survive, are less likely to be garbage and are more likely to be again required in the future.

Container hierarchies may be of any height, and there can be any number of independent hierarchies in the system. On attempting to retrieve an object from its logical container, the request is passed on to the parent container if the object is not found locally. The search ends when the object is found, in which case it is brought back to its original container, or the root is reached, in which case the object has been deleted and the reference is dangling. Human intervention is sought when dangling references are detected to stale objects. The implementation of the scheme requires that information be gathered relating to the frequency of use of each object. The advantage of the proposal over garbage collection is the reduction of inter-container dependencies. Further the information gathered by the system does not need to be completely accurate, unlike that required to support the absolute guarantees provided by garbage collectors.

Two classes of objects require a special treatment. Some objects may be marked as permanent, so that they are never deleted, no matter how aged they become. On the other extreme, temporary objects, such as intermediate files generated by a compiler, can be marked as immediately disposable.

Two further functions provided by the storage system are *replication* and *versioning*. Replication is motivated by safety and efficiency considerations. A container is a logical unit of storage, which is implemented by one or more physical units such as disk volumes or disk partitions, usually at different sites (this concept is borrowed from Locus [Walker 83]). Some of the objects of a container need not necessarily be replicated. The SS is responsible for maintaining a (weak) consistency between the replicas of an object. There is an obvious interaction between the replication and ageing mechanisms - as objects age, they may be replicated less.

Versioning is the ability to generate a new state of an object from an existing one and to provide an independent reference to each state. A component of the SS, the version manager, is responsible for mapping version names to LLIs, maintaining the relations between versions (version graph), and retrieving the appropriate version of an object based on given criteria (name, recency of update and relations to other versions).

4.4 - The Transaction Manager (TM)

The Transaction Manager (TM) is responsible for implementing atomic transaction support. It is invoked when an atomic object is referenced in Virtual Object Memory, and when there are explicitly begin, end or abort atomic transactions, and checkpoints.

The TM uses locking for concurrency control on atomic objects. Various recovery mechanisms are being considered. An atomic transaction is represented by an object: the LLI of this object is the transaction identifier, and the state of the object is the set of locks currently held by the transaction. Nested transaction are supported, as are distributed checkpoints over which locks can be maintained. Checkpoints are particularly useful for supporting persistent control.

The TM consists of four main components: the lock manager, which manages the set of locks for each transaction; the recovery manager, which implements the restoration of a consistent state to the SS when necessary, and restart from a checkpoint; the checkpoint manager, which co-ordinates the capture of a distributed checkpoint, and the termination manager, which co-ordinates the termination (successful or otherwise) of an atomic transaction.

4.5. - The Type Manager (TPM)

The Type Manager (TPM) is responsible for maintaining type definitions, maintaining information on relationships between types, and making this information available during compilation, when an application is actually running, or during query processing (cf section 4.6). The Type Manager is supported by the Storage SubSystem.

Information to be provided by the TPM to compilers includes type definitions, and the subtyping relationships. The TPM also provides type representation information (in the form of access methods) to the Query Processor which uses it to perform associative accesses to objects in classes. Thus the TPM has two inter-related components: the first one deals with abstract types and type hierarchies, and the second is in charge of object representations and access methods.

4.6. - The Query Processor (QP)

The Query Processor (QP) supports the query language of Comandos. It is internally composed of four modules: placement manager, parser, query optimizer, and query execution manager.

The placement manager maintains information on the locations of objects in classes. It uses the Storage SubSystem to manage the placement and replication of instances, and the Type Manager to collect information about type definitions and type hierarchies.

The parser translates the query into an internal format (parse tree) and verifies that the query has the correct syntax.

The query optimizer has the task of deciding the execution strategy for a given query. It interacts with the placement manager, and the Type Manager to fetch information about type data statistics and access mechanisms. The query optimizer performs query decomposition in terms of the access operations defined on objects (search, update, etc).

Finally, the query execution manager has the task of executing the query according to the strategy defined by the optimizer.

4.7. - The Activity Manager (AM)

The Activity Manager (AM) supports the abstractions of job and activity which are the basic units of the computational model (cf section 3.1). It also support run-time mechanisms such as triggers, exception handling, and synchronisation primitives.

The Activity Manager is located at a low-level in the system; it has the form of a "distributed kernel", which implements the basic types needed to support the computational model. Types *Job*, *Activity*, *Semaphore* and *Trigger* and *Timer* are supported at each node of the system. Instances of these types contain in their instance data the information needed for their management (e.g. which sites a job has visited). Activities are locally supported by *Processes*, implemented on each site by a process manager, which performs the usual operations of process creation, deletion, and scheduling.

4.8. - The Communication Manager (CM)

The Communication Manager is responsible for the forwarding of communications to remote sites and for routing incoming requests to the appropriate service manager. It provides services upto and including OSI layer 4. Connectionless interactions are normally used at this level, although connections may be built to support the higher-level channels of the computational model. Jointly the Communication Manager and Activity Manger provide a Remote Execution Service to the other components of the Architecture. The Communication Manager also interacts with the Transaction Manager so as to support distributed commitment and checkpointing.

5. - Implementation Plans

Some aspects (the *kernel*) of the Comandos architecture are machine and host environment dependent, while others (the *system services*) are intended to run on any machine hosting the Comandos kernel. The present implementation plans by the Comandos consortium envisage three prototype implementations of the kernel. The first will be based as a guest layer on top of Unix System V for SPS7, Sun 2 and 3, and 3B series. Both of the other two implementations are for "bare" machines, and differ in whether multi-user environments can be supported or not. The second implementation is targeted for microVax-II and NS32000 series machines, while the third is for simpler single-user machines based on i286 and i386 processors. Such smaller machines may be used as satellites to access more sophisticated services hosted elsewhere. The chief implementation language is expected to be Modula-2, and UNIX will be used throughout as the development environment.

6. - Relationship to other projects

Within the Comandos consortium there are close contacts with several other ESPRIT projects due to common partners. The office system Document Server projects Multos (no. 28) and Doeois (no. 231) are both building sophisticated multi-media document servers based on the ISO ODA. While Multos is concentrating on storage and retrieval aspects, Doeois has focussed on support for distributed and interworking servers, and also for automated Office Procedures. Both projects are potential prototype applications for the Comandos infrastructure. The Somiw (no. 367) project is building an Object-Oriented distributed kernel for an Advanced Office System workstation, based on C++ and the Sun-3. Somiw is concentrating on integration of a full set of facilities for the user interface (including voice), rather than an integrated persistent distributed infrastructure for applications, as is Comandos. Nevertheless Somiw workstations may well be suitable Comandos hosts in the future.

Some other projects using an object approach include CSA (no. 237), Pool (no. 415/313) and to some extent PCTE (no. 32). CSA is perhaps the most relevent to Comandos since it is attempting to construct an architecture for Communicating Systems. Pool is building a massively parallel system based on the object approach and including special hardware support, rather than a loosely coupled distributed system based on conventional hardware as is Comandos. Nevertheless the development of the POOL-T language by Pool is being followed with interest. Finally although the PCTE OMS (Object Management System) provides a simple Object Store, the type system is weak and object sizes are comparatively large.

A further project of significance to Comandos is the Ansa project, partially sponsored under the UK Alvey programme. The object-oriented approach and computational model of Ansa for distributed systems are close to those of Comandos, although data management issues and language support are apparently currently outside the scope of the Ansa work.

7. - Conclusions

The Comandos project is building an infrastructure for the design, installation and operation of distributed environments for office systems and other applications. A major goal is the definition of an architecture to describe the operational and execution environment. This architecture offers an object oriented Virtual Machine for application programmers which allows both a common typing system for volatile and persistent objects, and a framework for distributed processing. The architecture is constructed from a number of functional components, which in themselves may be distributed. Prototypes of this architecture will be implemented by the consortium so as to verify the concepts before undertaking a full implementation.

References

- [Albano85] A. Albano, L. Cardelli & R. Orsini "Galileo: a strongly typed interactive conceptual language", *ACM TODS* vol 10, no 2, 1985.
- [Andrews87] G. Andrews et al. "The Design of the Saguaro Distributed Operating System", *IEEE Transactions on Software Engineering*, Vol SE-13, No 1, Jan 1987.
- [Baker87] Baker S. et al, "Office Procedures: a Formalism and Support Environment", *Proc. Esprit Conference 1987 (this volume)*, Brussels, Sept 1987.
- [Baron86] R. Baron et al., "Mach-1: Kernel Interface manual", Dept of Comp. Sc., *CMU Internal report* Jan 1986.
- [Black87] A. Black et al. "Distribution and Abstract Types in Emerald", *IEEE Transactions on Software Engineering*, Vol SE-13, No 1, Jan 1987.
- [Cheriton84] D. Cheriton "The V Kernel: a software base for distributed systems", *IEEE Software*, Vol 1, No 2, April 1984.
- [Dahl68] O. Dahl, B. Myhrhaug & K. Nygaard "Simula67 Common Base Language", *Norwegian Computing Centre*, Oslo, Norway, June 1968.
- [DASE86] "DASE Model", *ECMA TC32-TG2 Working paper 186/69*, October 1986.
- [deCouchant86] D. DeCouchant "Design of a distributed object manager for the Smalltalk-80 system", *ACM Proc. OOPSLA*, Sept 1986.
- [Goldberg83] A. Goldberg and D. Robson "Smalltalk-80: The Language and Its Implementation", *Addison-Wesley*, 1983.
- [Horn87] C. Horn "Conformance, Genericity, Inheritance and Enhancement ", *Proc. ECOOP 87*, AFCET, Paris, June 1987.
- [Lieberman81] H. Lieberman "A preview of Act1", *AI Memo No 625*, MIT, Cambridge MA, USA, June 1981.
- [Lieberman 83] H. Lieberman and C. Hewitt "A real-time garbage collector based on the lifetime of objects", *Comm. ACM*, vol. 26, 6 June 1983
- [Liskov83] B. Liskov et al. "Guardians and Actions: Linguistic Support for robust, distributed programs", *ACM TOPLAS*, Vol 5, No 3, July 1983.
- [Maier86] D. Maier and J. Stein, "Development of an Object Oriented DBMS", *Proc. OOPSLA*, Sept 1986.
- [Moon86] D. Moon "Object-Oriented programming with *Flavors*", *ACM Proc. OOPSLA*, Sept 1986.
- [Ness87] Ness A., Reim F., Percy R., Dowler B., "An Architecture for System and User Management Tools in a Distributed Office System", *Proc. Esprit Conference 1987 (this volume)*, Brussels, Sept 1987.

- [Schwarz86] P. Schwarz et al. "Extensibility in the Starburst Database System", *Proc. Int. Workshop on Object-Oriented Database Systems*, IEEE/CS, Asilomar (sept 86)
- [Shapiro86] S. Shapiro, "Structure and Encapsulation in Distributed Systems: The Proxy Principle", *Proc. of 6th DCS*, Boston, May 1986.
- [Thatte 86] S.M. Thatte, "Persistent Memory, a storage architecture for object-oriented database systems", in *Proc. Int. Workshop on Object-Oriented Database Systems*, IEEE/CS, Asilomar (sept 86)
- [Walker 83] B. Walker et al, "The LOCUS distributed operating system", *Proc. Ninth Symp. on Operating Systems Principles*, *ACM Operating Systems Review*, vol. 17, 5 (dec.1983).

Project No. 834

AN ARCHITECTURE FOR SYSTEM AND USER MANAGEMENT TOOLS IN A DISTRIBUTED OFFICE SYSTEM

Andreas J. Ness, Friedemann Reim, Helmut Meitner

Universität Stuttgart and
Fraunhofer-Institut für Arbeitswirtschaft und Organisation
Stuttgart, Federal Republic of Germany

Richard A. Percy, Surinder S. Makh

International Computers Limited
Basingstoke, England

An architecture for system and user management tools in an object-oriented distributed office system is proposed that supports the paradigm of adaptation and incremental change of the system. System design and management is hierarchically decomposed into four basic tasks: DOS design, application programming, security management, and configuration and operational control. Each of these tasks can be regarded as a complex decision process loosely coupled with the others, passing requirements and capabilities. For the basic tasks an interactive knowledge-based decision support tool will be provided. A major feature of these tools is that they interact on-line with the productive application system they manage. The running system itself is used as a source of information for its design with the help of a distributed system observation facility. Design and configuration changes can be implemented with a distributed parameterization facility.

1. INTRODUCTION

The work reported here is part of ESPRIT project 834, Construction and Management of Distributed Office Systems (COMANDOS). The objectives of the project are to provide a flexible, reliable and easy to use environment for the development and the management of distributed applications in the office context. For this purpose, an object-oriented distributed operating and data management system; a system for the integration of preexisting applications (COMANDOS integration system, CIS); and tools for the design and the management of the system are currently being developed. This paper reports on the approach taken for the development of the design and management tools in COMANDOS and on the architecture of these tools. This architecture of the design and management tools, together with the architecture of the object-oriented operating and data management system (Horn and Krakowiak (1987)) and the architecture of CIS, is part of the global architecture of the COMANDOS system.

This work has been supported by the Commission of the European Communities under ESPRIT Project 834 (Construction and Management of Distributed Office Systems).

Large and complex distributed office systems (DOS) in a dynamic environment require a substantial amount of (re-)design and management during a life span of typically several years. This design and management ranges from planning to operational control of the DOS and is carried out by human managers assisted by tools. This assistance provided by the design and management tools can range from fully automatic in the case of system generation, for instance, to highly interactive in the case of a decision support system for DOS design.

Currently, several national and multinational activities are under way to establish frameworks and architectures for management of distributed systems (ECMA (1986a); ECMA (1986b); ANSA (1987); Sloman (1987)). Also, methods and tools are being built that support the analysis and the design of offices and of office information systems (Pernici and Vogel (1987); Schäfer et al. (1987); Zeviar (1985); Ness, Reim, Meitner and Niemeier (1986)). In this project, the aim is to establish an architecture of tools that integrates the (organizational) DOS design and the DOS management into an operational distributed office system such that it can adapt to a changing business environment.

2. PARADIGMS AND ASSUMPTIONS FOR THE SYSTEM DESIGN AND MANAGEMENT OF A DISTRIBUTED OFFICE SYSTEM

The approach to system design and management for a distributed office system is governed by the observation that a DOS "lives" in a dynamic office environment and that a formal life cycle-oriented model for design, management and maintenance appears less suited. The paradigms underlying the COMANDOS design and management approach are made explicit below.

- In most of the cases when a design or configuration activity takes place, the system is changed or upgraded rather than newly implemented. New installations can be regarded as special cases in this framework.
- The structure of the office and of the distributed office system has to adapt to the changing environments and needs, i.e. changes of the business requirements, upcoming new technology, or changes in tariffs. This has to be reflected in the design process and the architecture of the design and management tools.
- For the design process to be adaptive, analysis of the system and design has to be carried out in an incremental way.
- When a system is to be changed or upgraded, the existing distributed office system can be used as a source of information.
- The distributed office system should have domains and levels of control (cf. Sloman (1987)). The concept of a central system manager in the traditional sense who can do virtually everything does not exist.
- The DOS should have domains and levels of security. The concept of an overall security manager must exist for interdomain control but individual domains will have their security controlled by domain security managers. The overall security manager, however, does not automatically have all the power of a domain security manager.
- The DOS should have an overall security policy. Each security domain within the DOS has its individual security policy consistent with the overall policy.
- The system should operate as independently as possible of a human operator or system manager. In a domain of the DOS such as a particular office, several users applying the DOS for their work are present but no dedicated systems manager. The role of system manager is performed by one or several "end" users.

- The approach taken to define the tool elements and interfaces is a functional one. The mapping of the functions onto programs and human designers is carried out in a second step.

3. THE DESIGN AND MANAGEMENT PROCESSES

The four most important "creative" activities in the management of a large and complex distributed office system are its design, its configuration, its application programming and its security management. Extending the view taken in Ness, Reim, Meitner and Niemeier (1986) these activities can be understood as coupled decision processes whose relations are shown in Figure 1.

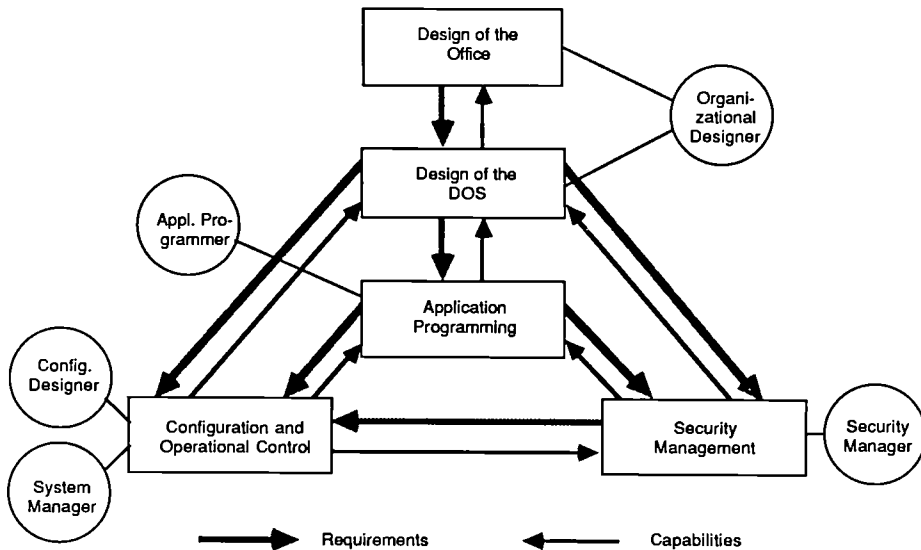


Figure 1: The Basic Tasks for the Management of a DOS

Two basic relations exist, the passing of requirements and the information about capabilities which must be taken into account. The design of the DOS passes requirements to the configuration indicating which functions must be provided at a certain location in the office, e.g. electronic mail, a C programming environment, a text processing facility and a certain user interface for a particular office worker. The configuration, in turn, provides the design of the DOS with information about possible performances and system structures which have to be taken into account for the design in order to arrive at a realizable design and pose realistic requirements.

Application programming, although an important activity in the design and management of a DOS, will not be further investigated here. COMANDOS deliberately does not concern itself with the process of application programming.

In the following, the processes of DOS design and DOS configuration will be examined in more detail. The approach taken toward security management is very similar to that of configuration, in particular, its relationship to DOS design.

3.1. The Design of the Distributed Office System

Figure 2 illustrates the environment of the design of the office and of the DOS, in particular, their relationships to the other basic tasks of Figure 1. For simplicity, only the relation to the configuration is shown. The relationships to security management and application programming are analogous.

The design process can be stimulated by an organizational redesign of the office, by measurements made in the running system pointing to the need to improve something, or by newly available technology or changed tariffs.

As mentioned above, the design of the DOS is understood as a decision process that is loosely coupled with organizational office design, with configuration, security management and application programming. The loose coupling implies that there is a bidirectional and concurrent information flow. This is significantly different from the view taken by phase-oriented models.

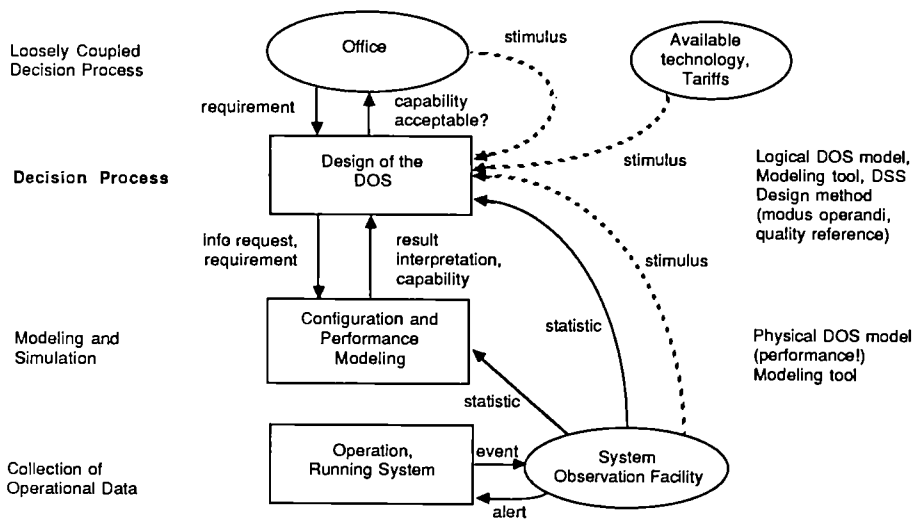


Figure 2: The Design of the Distributed Office System

The design of the DOS has to consider the requirements posed by the office environment which in turn checks the proposed solution for acceptability. The design of the DOS determines capabilities of a DOS that have to be taken into account when designing the organizational structure of the office. Since the two processes are loosely coupled in both directions there is a need to reconsider the organizational design of the office while designing the DOS.

Designing the DOS involves frequent evaluations of performance, cost and risks - based on the logical DOS model (see Section 5). For that purpose the design of the DOS activity will pose information requests to be answered by the configuration and security management stating the expectable capabilities of the proposed DOS. Both, the design of the DOS and the configuration can use data collected in the operation of the system as a source of information.

According to the design paradigms outlined in Section 2, change and adaptation rather than new implementation are of particular importance in the design process. This implies that DOS design is an incremental and iterative (generate and test) decision process. The most important functional components of this decision process that should be supported by a tool are the generation and the assessment of a solution (Ness and Reim (1987)).

3.2. The Configuration of the Distributed Office System

The configuration and performance modeling proceed interacting with the other tasks as shown in Figure 3. To simplify the picture, the relations to the application programming and the security management have been left out. They are of the same nature as the one to the design of the DOS. It is important to point out that the stimulus for action can either come from the design of the DOS, changes in the available technology and tariffs (environment), or from the running system, the latter calling for system tuning.

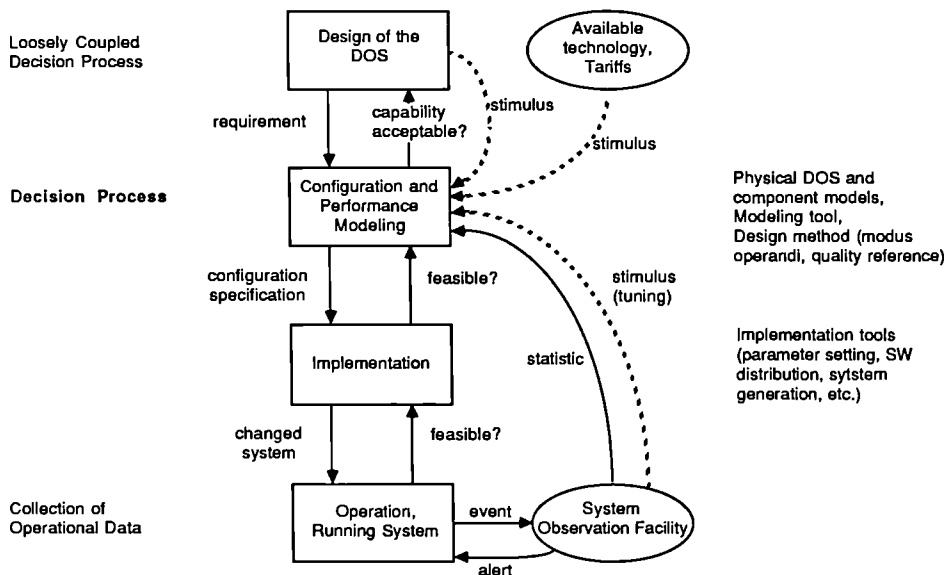


Figure 3: The Configuration of the Distributed Office System

For a more detailed discussion of the configuration process its last functional component, the implementation, is shown separately. This implementation includes procurement, installation and system generation. Configuration, implementation and operation are coupled in both directions. The backward coupling with the question of feasibility and usefulness must not be neglected. This backward coupling does not allow a purely sequential approach to the main tasks but rather calls for loosely coupled, interacting and partially concurrent activities.

4. THE INTERACTION OF THE DESIGN AND MANAGEMENT WITH THE RUNNING SYSTEM

As is shown in Figures 2 and 3, the design of the DOS, the configuration and also the security management use information obtained from the running system. This information is to be collected at various levels of the system, from the operating system to a particular application and at various locations in the DOS.

Two types of information can be distinguished: *alerts*, being events requiring immediate action of the intended recipient, and *statistics*, to be stored for later use. When observing a running system a large amount of data is generated and the above information has to be filtered from the irrelevant data.

The different types of information require that an "intelligent" filtering takes place immediately after the data are obtained in order to classify them and take the appropriate action. This could either be storing the data identified as statistic or passing an alert to a specific recipient. It can be expected that the distinction between a statistic and an alert depends on the environment and on the circumstances and is not entirely a function of the single piece of data itself. This suggests that the task of the "intelligent" filtering is nontrivial.

For the design, configuration and security management tasks three generic types of activities can be distinguished. The *observation* activities are concerned with collecting information while the system is running. The *activation* activities realize a design and management decision, i.e. implement a change. Clearly, these two types have to be on-line and integrated into the running system. The third group, the *decision* activities support the actual design, configuration and security management decisions. Modeling belongs to this group, for instance. Decision activities do not necessarily need to be on-line in the system. However, they need an information link to the observation and to the activation activities.

Section 5 will concentrate on the decision activities while Section 6 will address the observation and activation activities.

5. DESCRIPTION OF THE DESIGN AND MANAGEMENT TOOLS

A common framework for these tools will be outlined first, followed by the descriptions of the individual tools for DOS design and configuration.

5.1. The Modeling Context of the Design and Management Tools

Figure 4 shows the modeling context of the design and management tools. Three abstraction levels have been distinguished: The *logical DOS model* level for the design of the DOS is tightly linked to the organizational design of the office and the DOS. It comprises an office and DOS model with which office work can be described and the DOS can be represented in terms related to office work from a user perspective.

The second modeling level, the *physical DOS model* represents the DOS in concepts and terms suitable for determining the most appropriate system configuration, i.e. it is a descriptive and a performance model of the distributed office system. This model belongs to the configuration activity and considers the complete implementation of the DOS. Clearly, this physical model is very close to the actual implemented system. The relationship between the logical DOS model and the physical DOS model is a complex mapping of

a DOS representation close to the terms of the office work onto a physical system configuration.

The third modeling level, *the component model*, refines the physical model along an aggregation relation and considers single components of the system. The concepts and terms of the component model are not different in nature from the physical DOS model. The component model is merely a more detailed and modular representation of the physical DOS model. The component level is associated with the configuration activity.

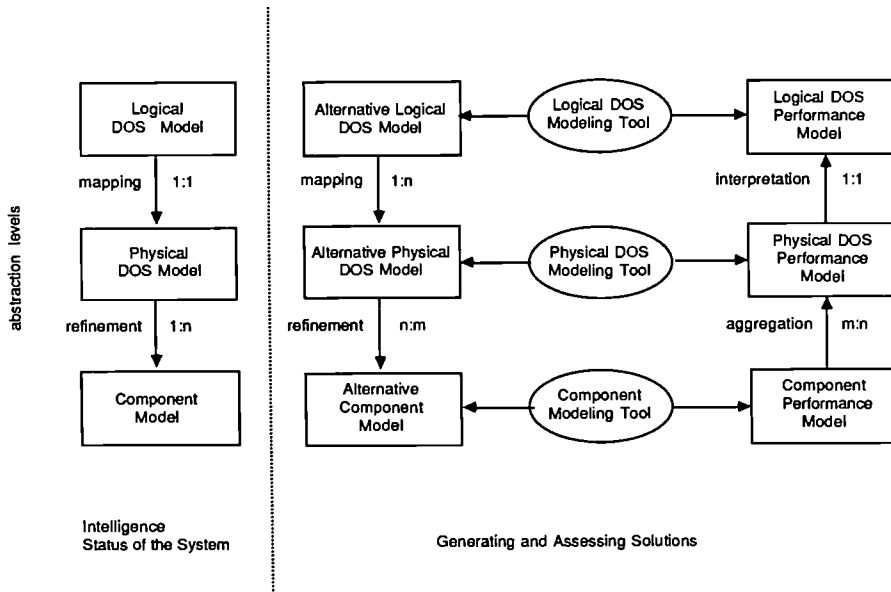


Figure 4: The Modeling Context for the Design of a Distributed Office System

For each abstraction level shown in Figure 4 a distinction can be made into the unique model of the current running system established with information collected by tools in the running system, and into models used in the process of generating and assessing alternative solutions. Within the latter group two types of models can be found. A descriptive model represented in the same formalism as the model of the running system and a assessment-oriented model relating a design to performance, cost and risks.

The most important function in this context, here associated with a modeling tool, is relating a descriptive model showing what the system or a proposed alternative looks like to the performance model showing how the respective performance, cost and risks will be. This function appears on all three levels and is supported by the respective tool.

What is not explicitly mapped into a tool in Figure 4 but can be derived from Figures 2 and 3 are the functions of relating the information contained in the models from one level to another, e.g. interpreting performance data from the physical DOS model at the logical DOS model level. This will be addressed in the subsequent sections with the description of the tools.

5.2. The Tool for the Design of the Distributed Office System

The tool for the design of the DOS is a decision support system (DSS) for the organizational designer whose responsibility is to select the components appropriate to support the office tasks, and the conceptual specification of the hardware and software system (cf. Figure 1). Central to the DOS design tool will be a logical office and DOS model that describe the actual running system and the current state of a design solution. It also has to represent information about expected performance, cost and risks of a design solution. Figure 5 shows the basic structure of the tool.

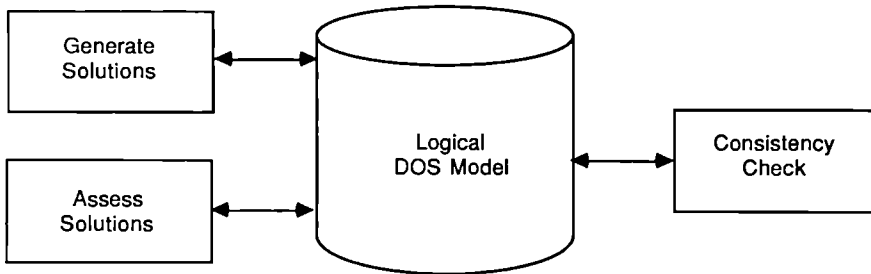


Figure 5: The DOS Design Tool

5.2.1. The Inputs and Outputs of the Design of a Distributed Office System

For the design of the DOS six inputs and six outputs have been identified:

The *design knowledge* comprises design knowledge of the kind "how to". This knowledge is necessary for the design of a DOS in general. The design knowledge can be refined along the aggregation relation (Figure 6). The *situation specific design knowledge* could be a specific, problem-oriented algorithm, e.g. for minimizing the duration of an office process. The *knowledge specific to the actual office* is a descriptive representation of the actual office. It may include information on the hardware and software available and on the people working in the office under consideration. The *application area specific knowledge* includes the knowledge on the relevant area, e.g. a secretarial environment or production planning. The *general knowledge about DOS* comprises information available about distributed office systems that is considered relevant for their design.

The *situational requirements* as opposed to the design knowledge add the specific features of the design situation. The requirements state "what to do", i.e. the specific problem to be solved, but not "how to do it". This could be to reduce the duration of a particular office process, for instance.

The *status and history of DOS and office* are gained by the observation activities in the running system. Status and history are not restricted to information observed in the running DOS. Information out of the office itself will also be included. Part of this information will be collected manually by humans.

The remaining input reflects the capabilities of the underlying basic tasks (cf. Figure 1). The knowledge on the actual implementation of the design, e.g. parameters, resulting from the respective basic tasks is included here. The *security management capabilities re-*

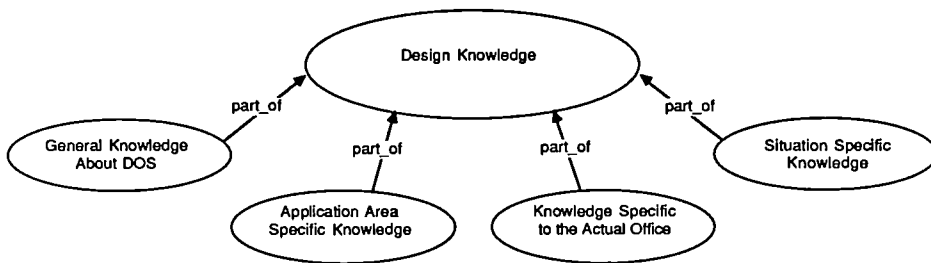


Figure 6: Design Knowledge for the Design of a DOS

present the information on the capabilities of the actual implementation and on the potential capabilities of the security management. The *configuration capabilities* do the same for the configuration. The *application programming capabilities* reflect the capabilities that are provided by the application programming, of course.

The "productive" outputs *requirements for security management, configuration, and application programming* propagate the respective needs/inquiries to the cooperating tasks (cf. Figure 2). An interface to the respective tasks will be defined.

The *DOS capabilities* have to be available for usage by the design of the office in a way similar to how the DOS design relies on the capabilities of the other cooperating tasks. The *design process log* keeps track of the design steps that have been carried out. It is useful as a simple audit facility but it also can be used for meta-activities such as the installation of new design rules. *Activation* directly triggers activities to be carried out in the running system.

5.2.2. The Interfaces of the Design Tool

In order to support the DOS design process the DOS design tool needs interfaces to the organizational designer, the running distributed office system, the configuration, the security management, and the application programming (Figure 7).

Requirements and capabilities must be communicated to and from the other basic tools. Therefore, specialized interfaces with capability models and requirement models will be provided.

As shown in Figure 2 the DOS design obtains information from the running system in form of statistics that are collected by the on-line observation facility. Hence, the DOS design tool must foresee an observation interface.

In the other direction an activation interface will exist that implements DOS design decisions in the running system, e.g. changing system parameters.

Another very important interface of the design tool is the one to the organizational designer. Figure 7 shows two different interfaces, one for solving and one for modifying. In contrast to the other interfaces these are bidirectional. Solving and modifying designate two different modes of interaction between the DOS Design Tool and the organizational designer. In the former the organizational designer's objective is to solve a particular design problem. In the latter the organizational designer intends to change or modify the DOS design tool itself. This can be expected to occur frequently when new design know-

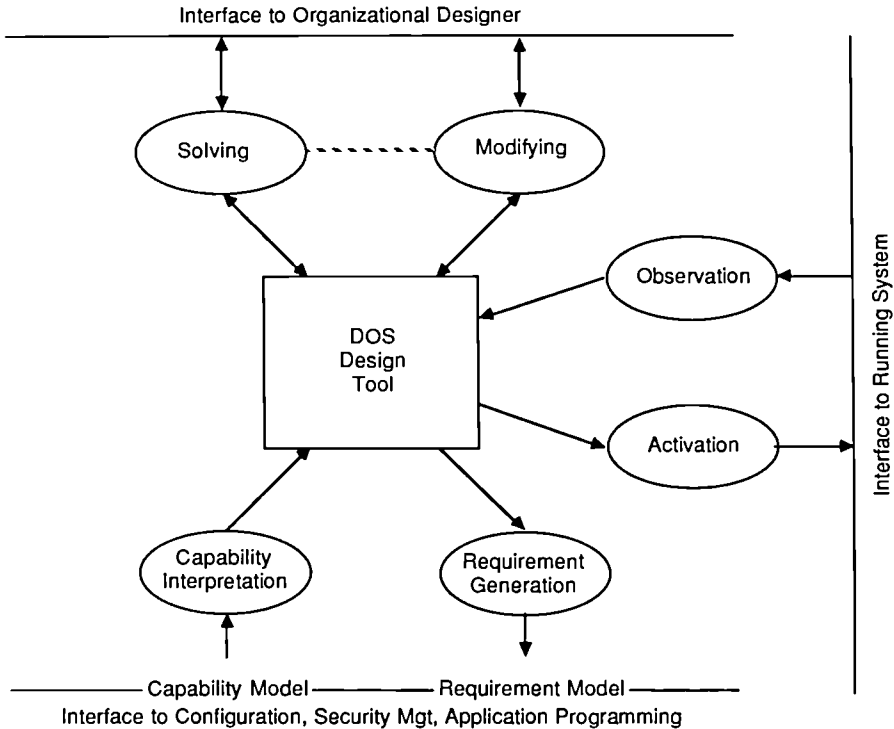


Figure 7: Interfaces of the DOS Design Tool

ledge arises or new problem areas have to be dealt with. In practice these two interaction modes can occur in the same session, e.g. when the organizational designer solves a problem and encounters an inadequacy in the DOS design tool he may immediately modify the tool.

5.2.3. The Functionality of the Models and the Tool

The DOS design tool should support a human organizational designer in the following tasks:

- Representation and generation of a model of an office and its distributed office system.
- Derivation of DOS functionality required. This functionality is derived from a description of the office work in terms of a model.
- Determination of the structure and the necessary components of a DOS to support this functionality.
- Assessment of the performance, cost and risk of proposed alternative solutions to a DOS design problem.
- Information collection in the running distributed office system as a basis for design decisions.
- Implementation of design decisions either directly in the running system or through the other basic tasks of Figure 1.

The DOS design tool will be implemented on a Unix workstation with a high resolution graphics screen and a mouse. X-windows, Common Lisp and C will be used for implementation. It is intended to port it later onto the emerging COMANDOS system.

5.3. The Performance Modeling Tool for a Distributed Office System

The distributed office system for which a performance model is to be developed is comprised of a number of geographically distributed sites.

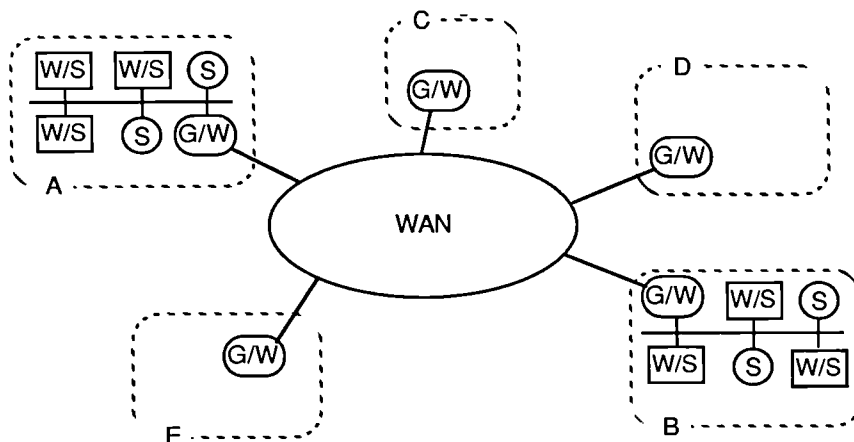


Figure 8: A DOS showing geographically distributed sites

Each site consists of a number of workstations (W/S) and servers (S) connected together via a local area network (LAN). Sites are interconnected through gateways (G/W) by a wide area network (WAN). This is shown schematically in Figure 8. A workstation at site A accessing a server locally or on another site B will use a particular route (Percy(1987)). In the latter case, the route would comprise traversing the LAN at A, gateway at A, WAN, gateway at B, LAN at B, server, LAN at B, gateway at B, WAN, gateway at A and LAN at A. A simpler sequence would occur if the server accessed is at site A. A more complex sequence results if the request is of the following kind: Opening a client/server session, which typically generates individual exchanges between a user node and a directory server (or set of directory servers); between the user node and an authorization server; and between the user node and the target server node.

The model to be developed identifies the queuing centers along routes. For each center it will assume a Poisson inter-arrival rate for jobs and an exponential service rate. In queuing terminology this is known as an M/M/1 queue. This type of queue has been analyzed extensively in the literature. For each queuing center a response time can be calculated. This response time is the time taken to process the request and includes both, queue time and service time. The model will then sum the service times for each center along the route to obtain the total response time for a particular transaction.

A queuing center can itself be broken down further if required, e.g. a mainframe server. Thus a model could be developed for the subsystem. The results for this subsystem (for time, response, utilization etc.) can be input back into the overall model. Hence the ad-

vantage of this approach is that it allows the addition of new models to the overall modeling scheme as new types of subsystems are introduced.

The goal of the model is to provide reliable performance estimates of component utilization and response time. The underlying accuracy of the model is determined by the accuracy of both, the source data and the queuing algorithms. It is not the intention of the model to give "absolute" predictions but rather to provide accurate indications. In real operation the COMANDOS system will be subject to varying levels of activity and at any time the utilization and/or response time can vary markedly even for the same transaction. Therefore it is considered to be only of limited value to know the "absolute" values. Consequently a quality target of $\pm 10\%$ of actuals is envisaged as being a suitable target.

5.4. The Tool for the Configuration of the Distributed Office System

5.4.1. Overview

Details about the data-systems, communications services and distributed applications in the network community are held in a database model of the network in the form of the configuration network model. This model is based on the OSI Basic Reference Model. The elements that make up the model are shown in Table 1.

Service Component	A component of a service at a particular layer (e.g. Transport Service) in a single data system.
Service Access Point (SAP)	The point through which a Service component accesses a Service component in the layer below.
Component Association	The logical link between two Service Components in the same layer that provide a connection.
Interworking Requirement	The logical link between two SAPs that shows how a component association in one layer maps down into the layer below.

Table 1: Elements of the Configuration Model

The model is based on the concepts of layers, i.e. the associations between service components within a layer and the mapping of these associations onto the layer below. Service components within a layer are accessed from the layer above via SAPs. The interworking requirements between SAPs must be satisfied by the associations between service components in the same layer. Mapping these associations onto the layer below defines interworking requirements in that layer.

5.4.2. The Structure of the Configuration Generator Model

The purpose of the configuration details held in the database is to enable the administration center to send information to the various open systems in the network, allowing them to configure the data systems for which they are responsible. This requires a model of the communications services and distributed applications.

The model is based on layers which can be stacked up, one on top of another. Each layer is functionally independent of the layers above and below it. A layer provides a service to the layer above it and uses the service provided by the layer below it. This allows layers to be changed or deleted, or new layers to be added, without affecting the other layers. Within each layer there are a number of service components which are hosted on data-systems in the network. The service components in the community may be viewed "horizontally" within a layer or "vertically" above a node.

Within a layer, the service components form associations with each other. These associations represent the communication links between the service components. Associations in an upper layer are mapped onto the layer below. This mapping between layers establishes the vertical structure of service components hosted on a particular node. The associations between service components establish the horizontal structure of the layer across all nodes.

Service components within a layer are accessed from the layer above via SAPs. The mapping of associations from the layer above defines interworking requirements between the SAPs in this layer. These interworking requirements must be satisfied using associations between service components in the same layer.

6. THE INTEGRATION OF THE DESIGN AND MANAGEMENT TOOLS INTO THE RUNNING SYSTEM

6.1. Structure

The basic components which must be placed within the managed system are: generators, collectors, processing functions and system observation facility (SOF). The SOF for the system as a whole is made up of a number of system observation facilities for each of the subsystems comprising the system. A SOF is itself distributed. Forthcoming work within the COMANDOS project specifically require the implementation of an alerting capability and a statistics database. Both these facilities will be implemented from: submission agents; storage agents; and access agents.

A number of cases have been considered (Dawe and Percy (1987)). For example, all elements co-located, e.g. a large multi-purpose server, generators not co-located, e.g. when providing a "management station" for a number of domains, distribution of a SOF and SOF at different levels. Here, we will only consider the last case in any detail.

Figure 9 shows a pictorial representation of a SOF at different levels. Here G represents generators and C collectors of statistics. P is a processing function which acts as a filter or router of information, i.e. it decides which information to discard or record, and its routing.

Processing the data between collection and submission gives an opportunity to extract data from a subsystem for submission to a SOF at a higher level. For example, by extracting alerts from a number of different kinds of management stations, it becomes possible to submit to a "total system" management station, information from a number of subsystems within it which can then be used to resolve which subsystem is actually the source of a problem.

6.2. Interface Considerations

This section considers the interface definitions required to enable the submission of management information from a generator through a collector and processing function to a SOF.

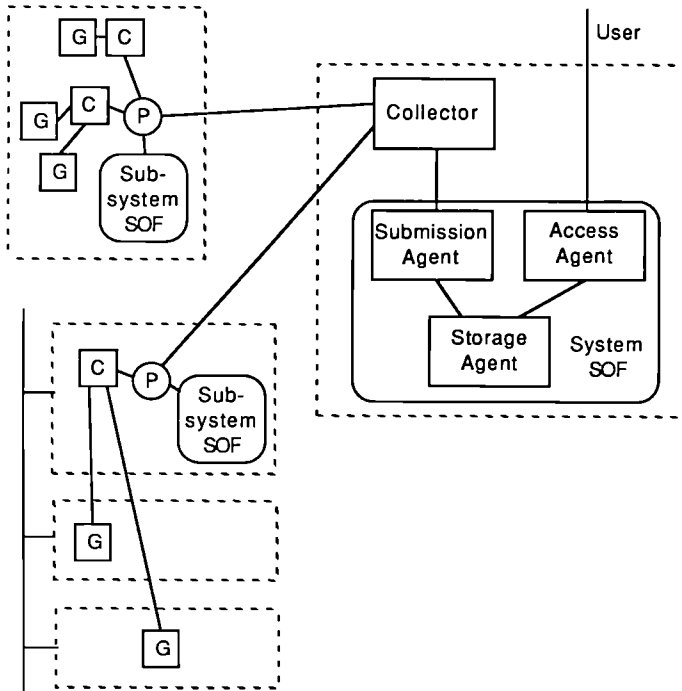


Figure 9: Levels of System Observation Facility

A collector may be required to monitor two basic types of distributed system entities. These are:

- Persistent generator entities
- Transient generator entities

Management must be able to address specified "persistent" entities of a distributed system e.g. the operating system, communications subsystem, etc., and may therefore actively exert control and "collect" statistics, as necessary. It must be noted that Dawe and Percy (1987) originally used the term "permanent" entity. However it is now felt that the term "persistent" entity is more appropriate as it conveys the fact that the entity may become temporarily inaccessible, i.e. out of service, for short periods, but is intended to usually be available to support specific systems facilities.

The transient entities of a distributed system, e.g. applications processes, communications connections etc., present a very different problem for management. Because of their dynamic nature, a transient entity is ultimately created and activated via a sponsoring persistent entity within the system, e.g. the operating system, communications protocol entity etc. It is therefore proposed that the active management of transient entities is always achieved via their sponsoring persistent entities. However, a "passive" mechanism must be provided to enable transient entities to asynchronously submit management information such as events and terminating statistics, e.g. status reporting, transaction counts, response times etc.

The discussion above has shown that there is a requirement for the definition of two generator-collector service interfaces. This is shown in Figure 10. This architecture is intended to provide a consistent management environment for the constituent components, irrespective of whether or not the actual collector is in the same node of the distributed system as the generator. Figure 10 also introduces the concept of a "collector agent" which provides a transparent "remote" path to/from the collector. The arrows indicate the nature of the communication between generators and collectors. That is between persistent entity and collector the communication is a two way process but only one way between transient entity and collector. This implies that management can only exert control over a transient entity via its host, the persistent entity.

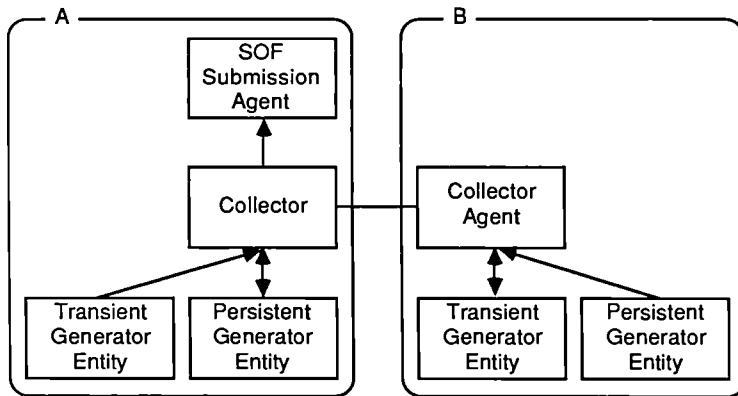


Figure 10: "Local" Collector and "Remote" Collector Agent

Future work is concerned with studying the parameters that have to be supplied at the submission interface between collector and SOF to enable the flow of information to appropriate aspects of the SOF.

7. CONCLUSIONS AND FUTURE WORK

With the emergence and increasing diffusion of distributed office systems design and management of these systems will become key factors for their successful use in a dynamic, rapidly changing office environment. COMANDOS proposes an approach to design and management of distributed office systems that is based upon the concepts of adaptation, hierarchical decomposition into loosely coupled decision processes, tools support, and integration into a running DOS, thus merging design, management and maintenance into a common framework.

After establishing the approach and the respective architecture of the tools the interfaces between these tools have to be defined. Work is currently in progress deriving the interface concepts and languages from the models and techniques used in the tools.

The architecture and the concepts of the emerging object-oriented distributed operating and data management system of COMANDOS have been specified recently. The most challenging work to come will be the integration of the approach and architecture of the design and management tools with this distributed operating and data management

system. One aspect will be the establishment of the detailed way, design and management will view and approach the objects of the DOS, i.e. which objects are managed and how. Another important task will be the integration of the design and management tools that directly interact with the running system, in particular, the system observation facility. The system observation facility is itself a distributed, hierarchically and geographically structured application of the object-oriented distributed operating and data management system.

ACKNOWLEDGEMENTS

The authors would like to thank Petra Hirsch (Universität Stuttgart), Joachim Niemeier (Fraunhofer Institut für Arbeitswirtschaft und Organisation), Peter Dawe (ICL), Nigel Ball (ICL) and Brian Dowler (ICL) for their contributions.

REFERENCES

- ANSA (Ed., 1987), ANSA Reference Manual, Release 00.02, Cambridge, England, May 1987
- Dawe, P.J. and R.A. Percy (1987), Modules for Implementing Instrumentation and Event Management, Interim Report, Deliverables D2 of Work Packages T3.3.4 and T3.3.8, ESPRIT Project 834 (COMANDOS), ICL Network Systems, Basingstoke, England
- ECMA (Ed., 1986a), Distributed Applications Support Environment (DASE), Overview and Services Provided, Draft 0.6, April 1986
- ECMA (Ed., 1986b), Framework for Distributed Office Applications, Final Draft, ECMA TC32-TG5, December 1986
- Horn, C. and S. Krakowiak (1987), Object-Oriented Architecture for Distributed Office Systems, in this Volume
- Ness, A.J., F. Reim, H. Meitner, and J. Niemeier (1986), Decision Support System for Planning and Design of Distributed Office Systems, Deliverable FHG-D1-T1.1-860829 of ESPRIT Project 834 (COMANDOS), Universität Stuttgart and Fraunhofer-Institut für Arbeitswirtschaft und Organisation, Stuttgart
- Ness, A.J. and F. Reim (1987), Planungs- und Gestaltungswerkzeuge für verteilte Bürosysteme - Eine Übersicht und Ansatzpunkte für eine Methodenintegration, Online '87 in Hamburg, 4-7 February 1987, (in German)
- Percy, R.A. (1987), Data for Instrumentation and Event Management, and Event Management, Interim Report, Deliverables D1 of Work Packages T3.3.4 and T3.3.8, ESPRIT Project 834 (COMANDOS), ICL Network Systems, Basingstoke, England
- Pernici, B. and W. Vogel (1987), An Integrated Approach to OIS Development, In: ESPRIT '86: Results and Achievements, CEC 1987, p.835-844
- Schäfer, G., et al. (1987), Functional Analysis of Office Requirements, ESPRIT Project 56 (FAOR), Main Report
- Sloman, M. (Ed., 1987). Distributed Systems Management, Report, Dept. of Computing, Imperial College, London, 1 April 1987
- Zeviar, F.T. (1985), Distributed Intelligence in Alternatives Analysis for Computer Systems Selection and Configuration, In: Expert Systems and their Applications, 5th International Workshop, Avignon, 13-15 Mai 1985, Vol. I, p.101-120

Project No. 998

SECURITY DEVELOPMENT: TRADITIONS AND INNOVATIONS

S. Farrell, E. Stephens, F. Williams (a)
M. Fougere, G. Ruggiu, V. Sorel (b)

(a) C.O.P.S (Europe) Ltd.,
Hume House, Pembroke Road,
Ballsbridge, Dublin 4, Ireland

(b) Societe Bertin and CIE,
B.P. no. 3,
78373 Plaisir Cedex, France

Abstract

Traditionally, development of highly secure office systems has been based on a comprehensive analysis of threats. However, recent research indicates the desirability of starting by analysing needs for security and privacy.

Against a background of traditional threat analysis, we present two models, each representing a significant advance in development methodology for office systems security. The interplay between the models is shown by example and directions for the future are discussed.

1. INTRODUCTION

Traditionally, development of highly secure office systems has been based on a comprehensive analysis of threats. However, recent research indicates the desirability of taking a completely different approach to security development.

Against a background of traditional threat analysis, two models, each representing a significant advance in security development, are presented here. The first of these is called MoSel (Model for Security Development), and provides a general framework for security development from analysis of requirements through to implementation and acceptance of measures and procedures which ensure security and privacy. The second is a three-axis model for

cryptographic key management, which shows how specific key management structures should be tailored to the needs of particular applications.

The interplay between the models is shown and the advance over traditional approaches which they represent are noted.

2. SECURITY DEVELOPMENT BASED ON THREAT ANALYSIS

Traditionally, security development has been based on threat analysis. The general approach is based on the assumption that the first step towards a totally secure system is to identify all actual and possible threats.

Intellectual manageability of this task is brought about by classifying threats. In (CR and COPS, 1986), threats are classified according to whether they are

- deliberate or accidental
- active or passive
- physical or logical

For example, modification of the number of hours worked by an employee in order to increase pay is a logical, deliberate, active threat to a payroll system, while disclosure of an employee's basic pay by careless disposal of a hardcopy constitutes a logical, accidental, passive threat to the system.

Once threats have been identified, countermeasures to detect or prevent them must be developed. Development of countermeasures is based on risk analysis, which determines potential loss if a threat is realised, as well as on tradeoffs between facilities to detect and prevent realisation of threats.

3. MoSel: AN INNOVATIVE APPROACH TO SECURITY DEVELOPMENT

3.1. Background

Although threat analysis provides a good foundation for the implementation of measures to counter threats, our work in that area convinced us that requirements for security and privacy provide a more appropriate starting point for developing highly secure office systems.

Analysis of requirements for office systems security results in a clear statement of the need which must be satisfied by any measures to ensure security. Later, when a design for security has been created, the design is analysed with respect to the statement of requirements, exposing threats due to design. When a design has been implemented, and countermeasures and procedures have been put in place, yet another threat analysis should be carried out, to reveal threats due to faulty implementation of the security design.

This approach to the development of secure office systems is captured by MoSel, which describes the process of security development by analogy with software development.

MoSel represents a completely new approach to the design and development of security aspects of computer-based office systems. MoSel consists of a process model, which guides the activities involved in security development, and a product model, which provides a powerful medium for expressing the results of the process. The breadth of applicability of MoSel to various aspects of office systems security is indicated by its use in such diverse fields as communications systems, workstation security, and key management for cryptographic systems.

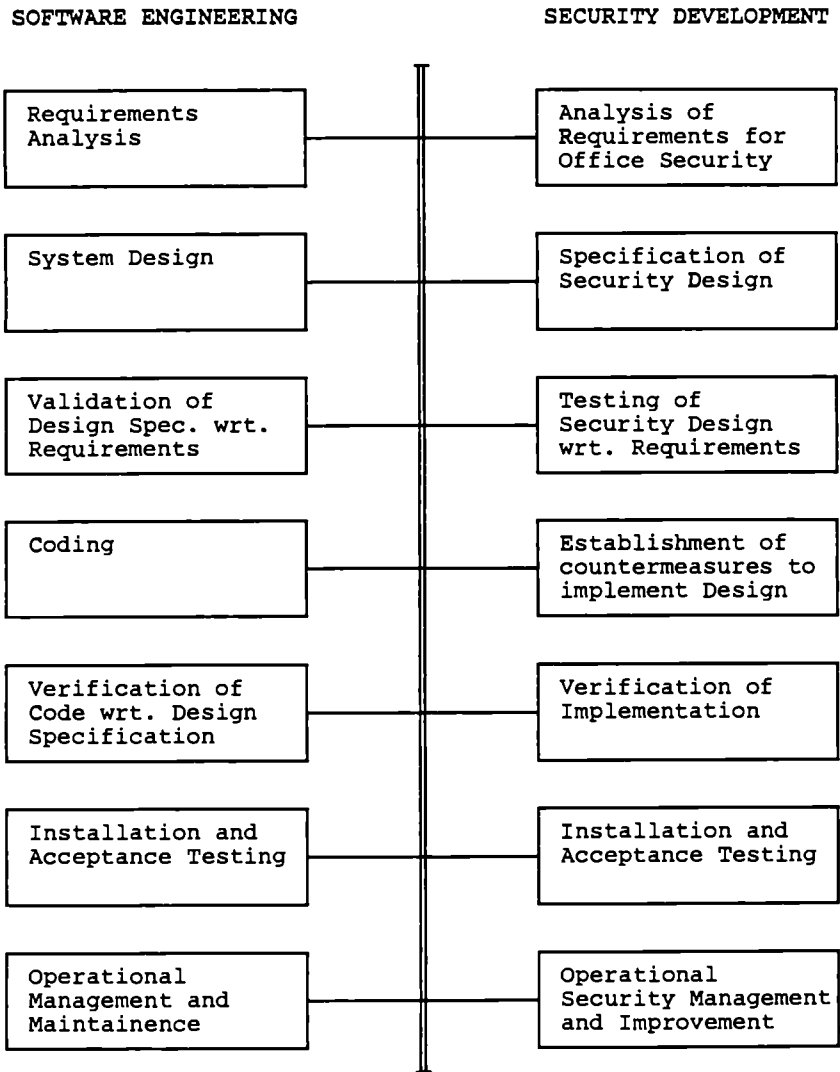
MoSel provides a sound methodological framework for security design and testing, and a practical medium for expressing the results of that activity.

3.2. The Process Model

Like software development, development of secure office systems is a process which leads from analysis of requirements for security and privacy, through design, implementation and user acceptance of measures to meet those requirements. The MoSel process model exploits this resemblance by describing the process of security development in terms more usually associated with software development. This analogy indicates how well-established methods for software development should be applied to security development, thereby enabling rapid evolution of security development methodology.

The analogy is also compatible with the view (Tomkins and Rice 1985) that security development and software development should take place together. By explicitly stating correspondences between activities in software and security development, the analogy identifies points at which software developers and security developers should liaise in order to ensure harmony between the various aspects of the total system.

The analogy is illustrated as follows:



Analogous Activities in Software Engineering and Security Development.

The main advantage of the analogy is that it enables the application of theories and strategies for software development to the development of office systems security. The significance of this lies in the fact that development methodologies for computer systems have advanced to a greater degree than development methodologies for developing secure office systems.

For example, in the context of the analogy, threats are seen as tests of the system. Methodology currently applied to the design of software test plans can be used to design threat plans to test the security of an office system. Moreover, a test plan methodology can be used to produce an enumeration of threats to a system, thus enabling application of established statistical quality control techniques to the development of highly secure office systems.

If a threat successfully penetrates the security of an office system, the implementation, design or requirements of the system must be reviewed and a flaw corrected or the design changed. This is analogous to a successful test which detects a defect in a software system which must then be debugged. Moreover, successful threats result in loss, as do system failures due to bugs. Potential losses, or vulnerabilities, are thus analogous to bugs in a system and can be classified as such.

The heart of the analogy is summarised as follows:

Threats	=	Tests
Vulnerability	=	Bug
Loss	=	Consequence of successful test

The interaction between threats can now be treated by considering combinations of tests. Future research on the process model will, in part, be directed towards the establishment of systematic methods for investigating threats and their interactions by analogy with established software testing methods.

3.3. The Product Model

The product model provides a powerful medium for expressing the results of security design and testing. It takes into account the human, hardware, software and temporal factors which influence security and privacy of office systems. The product model is essentially a language designed to express security aspects of computer-based systems. Each instance of the product model is a semiformal expression of a security design which is testable with respect to stated security requirements for a computer-based office system.

A detailed presentation of the product model, with a substantial example demonstrating its application, is contained in (COPS, 1987).

A case study using MoSel for designing the security aspects of workstations (CIRU, 1987) indicates its general applicability to security design.

However, to date the most significant indication of the breath of

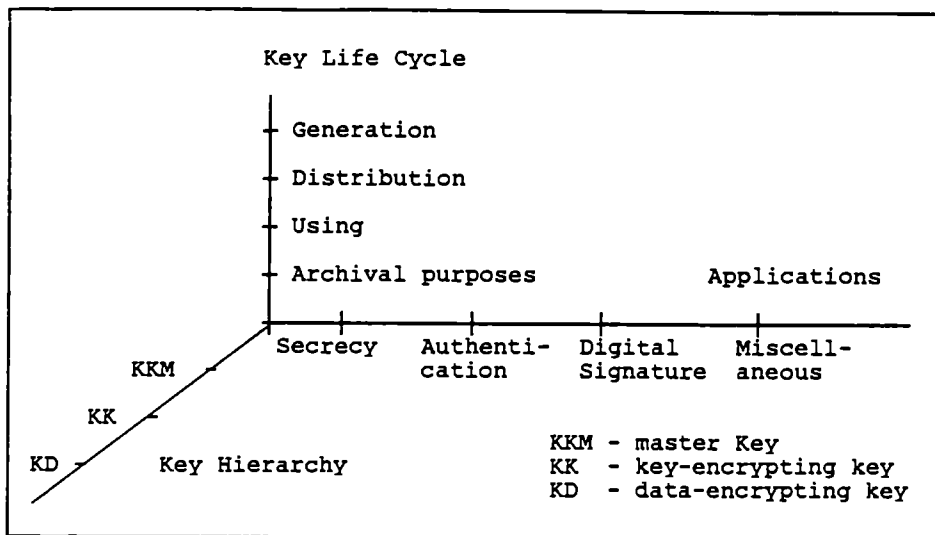
applicability of MoSel is its use in key management for cryptographic systems. The following section presents a novel approach to key management, whose relationship with MoSel is demonstrated later.

4. KEY MANAGEMENT FOR CRYPTOGRAPHIC SYSTEMS

4.1. A Three-axis Model for Key Management

Key management for cryptographic systems presents a serious problem to those who manage and develop highly secure office systems. A three-axis model for key management for cryptographic systems is represented by the diagram below. The axes of the model represent the three issues of secure key management.

Those issues are applications, key hierarchies and key life cycles. For each application of cryptography to office systems security, a specific key hierarchy, and a specific key life cycle provides a suitable key management structure. This model thus treats key management as an application driven activity.



A Three-axis Model for Key Management

4.2. Applications

Applications, which form a basis for the study of key management, lie along the first axis of the model. Applications include:

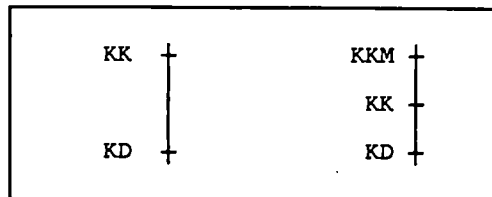
- secrecy, or maintenance of confidentiality of

transmitted messages both within an office and between geographically separate offices

- authentication, which combines identification of the sender of a message, and protection against data loss and modification of a message by an intruder, who might, for example, interfere with company accounts in an office system
- digital signature, or protection against repudiation and forgery which is particularly important for office applications involving transfer of wealth
- miscellaneous applications, such as password protection in the multi-user computing systems frequently found in office environments
- Within the context of the key management model, the application drives the development of the most suitable key life cycle and the key hierarchy.

4.3. The Key Hierarchy

To ensure a high level of security in a cryptographic system, the key used to encrypt the actual data to be protected must be changed frequently. Frequent changes necessitate frequent key transmissions between users. Depending on the application, these transmissions may themselves have to be encrypted. This introduces the concept of a key hierarchy. Two and three level key hierarchies are illustrated by the following diagram:



Two and Three Level Key Hierarchies

The Key-Encrypting Key, denoted KK, protects other keys, while the Data-Encrypting Key, denoted KD, protects data.

Data-encryption keys are used to protect transmitted or stored information. This key is also referred to as the Session or Working Key. The use of this key can be avoided by the use of Message Key within the encryption process. (The message key is called the Initialisation Vector, IV, in the DES.)

Key-encryption can be used to encipher either KD or another KK. The highest key level is called the Master Key, denoted KKM. KKM require a level of protection which is consistent with their level in the hierarchy.

4.4. The Key Life Cycle

The Key Life Cycle (KLC) is the third axis of our model. Along this axis lie the different phases in the life of the cryptographic key. These are:

- generation,
- distribution,
- usage,
- archival purposes.

Key generation usually provides random sequences of bits. It requires a secure cryptographic facility in which the plaintext of the key can be displayed.

The random generation of the keys can be achieved either by tossing coins or by throwing dice. Pseudo-random generators use an initial random value to produce sequences of values in which knowledge of one value does not provide information about other values in the sequence. An example of pseudo-random generation is the use of the DES algorithm to generate random keys for re use by the DES algorithm (Smid, 1981).

Master keys are generated in plain text, working keys in cipher text.

For asymmetric ciphers, keys of one or two-hundred digits in length must be generated - a time-consuming computation. The keys are large prime numbers, chosen randomly; probabilistic testing (Rivest et al., 1978) is proposed in order to determine the primality of numbers.

Distribution can be divided into several phases: sending, transmission and testing. The procedure can be manual or partially automated. Keys are generally transmitted through a key distribution network (KDN) which is either a dedicated network or a data communications network. This difference between a dedicated and a non-dedicated network must be taken into account when the security of the system is being analysed. The architecture of the KDN can be centralised, decentralised or distributed.

Before the transmission of keys, a secure back-up procedure must be implemented in the key generation area.

Manual distribution of keys is required for keys being transmitted in plaintext, with the knowledge of the key being split into two parts: one courier brings part of the key and a second courier brings the second part of the key. The two pieces are separately entered in the cryptographic facility under dual control.

Automatic distribution requires a logical communication network between users, providing a fast distribution system for the keys.

In a decentralised environment, referred to as a point-to-point environment (X9.17, 1985). The network is not included in the security analysis. Some of the users have key generation or acquisition facilities. Keys are distributed to each node on the network. If n users are present, $n(n-1)/2$ matched key pairs must have been arranged and each node must store $n-1$ of them. The

number of stored keys can be very large for an important network, but this architecture has the advantage that if one node is compromised, the other nodes remain secure.

In a centralised environment, distribution is assured by a Key Distribution Centre (KDC), which is acknowledged as trusted by the users. Key transmissions between nodes and the KDC are encrypted with a KK. Each node stores only the key shared with the KDC. Thus the number of keys stored locally is reduced but the security of the entire network is dependent on the security of the KDC. This system requires sophisticated protocols in order to ensure authentication and secrecy during communication.

Many protocols based on the use of secret keys have been published (X9.17, 1985; Abbruscato, 1984; Meyer et al., 1982; Needham et al., 1978). Others, based on asymmetric ciphers use published work by Diffie and Hellman (Diffie et al., 1976; Ingemarsson, 1982) can also be used for key distribution. Hybrid systems are based on the use of DES (Addendum to ISO 7498) for secrecy and on the RSA (Rivest et al., 1978) for authentication.

Asymmetric ciphers need directory management for distribution of public keys. Authentication is required for key transmission.

A distributed environment may be used for distribution as follows: Distribution is locally centralised in a hierarchical architecture with three levels: local, regional and global. Each controller has responsibility for the distribution of keys to the controllers on the level directly below them. The network can be easily extended and the number of keys stored on each node is relatively low.

Once the key has been distributed, it must be entered, stored, used and, if necessary, destroyed in a secure manner. This implies the frequent testing of physically secure facilities containing the cipher system. Usage is controlled by physical or logical control managers.

In some instances, keys must be archived after their phase of normal usage has ended. The life of the key must be longer than the life of the data that it protects. Procedures which permit recovery of the data should be developed in advance of the use of this system.

5. INTERPLAY BETWEEN MoSel AND THE KEY MANAGEMENT MODEL

The interplay between the two models during will now be demonstrated with regard to payroll security - a common office systems application. Free (1987) states fundamental aspects of payroll security as follows:

"Payroll data, and in particular the amounts of money being paid to various employees, should be secure from the prying eyes of the potentially envious"

"there should be good audit trails listing all changes entered, including overtime hours, bonus payments and so on."

The MoSel process model indicates that requirements for security

should be analysed. The results of analysis for this example are summarised in terms of data to be protected and required protection.

The data to be protected consists of:

- a) data maintained in the payroll system, including basic pay, employee identity, tax free allowance and so on, together with
- b) transactions entered at the time of each run of the payroll, consisting of overtime, bonus payments &c.

The security requirements are

- a) that data which reveal amounts paid should be read only by authorised persons,
- b) that all transactions should be traceable to the operator who entered them,
- c) and that the authenticity of transactions should be assured.

The MoSel process model indicates that measures to meet those requirements should be designed. The resulting design decisions are:

- a) to encrypt all data relating to the amounts paid to different employees
- b) to record all transactions in a Transaction File (TF) together with a digital signature identifying the operator and the time of entry of the transaction (c.f. Wood, 1982)

Further development of this initial design is guided by the three-axis key management model (KM3). For the purposes of demonstrating use of KM3, attention will be focused on protection of the transaction file. The points of interest on the applications axis of KM3, are secrecy and digital signature.

Secrecy refers to the requirement that some of the data involved in transactions should be encrypted (e.g. a transaction could be: "INCREASE FRED's SALARY FROM =!z^G2@af:: to 00£N1/4 Rxa"). For simplicity we assume that the whole TF is encrypted under K(TF).

Digital signatures will be used to record the operator and time of entry for each transaction. The operator's digital signature will be generated using a public and private key, where the latter is known only to an individual operator.

TF's are archived after each run of the payroll system. Thus, it is necessary to store the keys associated with each archived TF. Obviously, we need some mechanism to manage and secure these archived keys.

The second and third axes of KM3 refer respectively to the most suitable key hierarchy and key lifecycle for an application. A three level key hierarchy will be used for preserving secrecy of the transaction file. A key-encrypting-key (KK) will encrypt data

encrypting keys, in this case, the K(TF)'s. The life-cycle of the K(TF)'s is generation (for each run of the system), usage (during one run) and archival (with the KK used for encrypting archived key's). Finally KK's are stored encrypted under a master key (which is generated in cleartext and stored in a secure area).

The life-cycle of the keys involved in an operator's digital signature are more complicated due to the fact that new operators may arrive and old ones may leave. An operator who has left the company must no longer be considered valid but his public key (at least) must be retained for authenticating transactions stored in archived files.

At this stage in the development of a secure payroll system, MoSel indicates the need to analyse this design with respect to the stated requirements. This reveals threats to the key management system, which, when treated as tests, expose vulnerabilities due to design. Such vulnerabilities should then be corrected by suitable redesign of the key hierarchy and the key life cycle (following KM3).

For example, the above design makes it possible for an operator who has access to a K(TF) to decrypt an archived TF, enter a new transaction and re-encrypt the TF. To prevent this we modify the design so as to include an authentication code (like a MAC in communications) in encrypted files. This involves the use of a new key which can be managed in the same manner as the K(TF)'s.

Once a satisfactory design has been developed, it should be implemented, and the implementation should be tested to expose vulnerabilities due to implementation.

Finally, KM3 should be used to provide full documentation of the finalised key-management design, while MoSel should be used to produce documentation of the processes which lead to that design.

6. CONCLUSIONS

Together, MoSel and KM3 provide powerful support during the development of security aspects of office systems. MoSel provides a sound methodological framework within which systematic design and testing of security aspects of office systems can take place, while KM3 guides development of cryptographic key management, which has many applications in office systems security.

Based on an analogy with software development, the MoSel process model represents a significant advance on the current state of the art in security development methodology. The MoSel product model provides a means of expressing the results of the design effort in terms of interacting elements whose defining characteristics, patterns of interaction and existence vary in time.

MoSel also indicates how security methodology can, by analogy with accepted approaches to software construction, advance to meet the urgent need for a systematic approach to security development.

KM3, the three axis model for cryptographic key management,

represents a significant advance in this difficult area. It provides an innovative framework for the development of secure key management systems, which is unique in its treatment of the concept of time in key management.

Interaction between the models in development of secure office systems. Together, these approaches provide a new insight into security and privacy which is particularly applicable to office systems security.

ACKNOWLEDGEMENTS

We would like to thank our colleagues on ESPRIT project 998 (MARS), for their comments and criticism during the course of this research.

Credit is due to Dawn McMahon and Jean Whyms of C.O.P.S. (Europe) Ltd., and to Peter F. Mortenson and Jens Hanson of Christian Rovsing A/S of 1984, whose work on threat analysis (CR and COPS, 1986) exposed the need for new approaches to security development.

This work was sponsored by ESPRIT project 998, MARS.

REFERENCES

- Abbruscato, C. R., (1984)
 Data Encryption Equipment
IEEE Comm. magazine, 22(6), pp 15-21,
 September, 1984
- Addendum to ISO 7498 on Security Architecture,
 ISO TC97/SC21/WG16.1.
- CIRU, (1986)
 Computer Industry Research Unit,
 University of East Anglia
Study of State-of-the-Art of Security Threats and
 Countermeasures in Office Information Systems
 ESPRIT Project 998, MARS, Report no. 1, vol. no. 4
 document no. UEA/REP/0001, Issue 3.0, Sept., 1986
- CIRU, (1987)
 Computer Industry Research Unit,
 University of East Anglia
Model of a Secure Workstation
 ESPRIT Project 998, MARS, Private communication,
 April, 1987
- COPS, (1987)
 C.O.P.S. (Europe) Ltd.,
Security Model for Communications Systems
 ESPRIT Project 998, MARS, Report no. 2, vol. no. 6
 document no. Task 2.4, Issue 1.0

- CR and COPS, (1986)
 Christian Rovsing A/S af 1984, COPS (Europe) Ltd.,
Study of the State-of-the-Art of Security Threats
 and Countermeasures in Office Information Systems
 ESPRIT Project 998, MARS, Report no. 1, vol. no. 10
 document no. ST/REP/0003, Issue 1.0, Nov., 1986
- Diffie, W., and Hellman, M.E.,(1976)
 New Directions in Cryptography.
IEEE, IT-22(6), pp.644-654, November, 1976.
- Free, J. (1987)
 Hell to Pay?
Micro News, 5(1), May, 1987
- Ingemarsson, I.,(1982)
 A Conference Key Distribution System
IEEE, IT-28,No. 5, pp. 714-720, September,
 1977.
- Meyer, C. H. and Matyas, S. M.,(1982)
Cryptography: a new Dimension in Computer Data
 Security
 John Wiley and Sons, New- York, USA, 1982.
- Needham, R., and Schroeded, M., (1978)
 Using Encryption for Authentication in Large
 Networks of Computers.
Communications of the ACM, 11(12),
 December, 1978.
- Rivest, R.L. Shamir, A., Adleman, L., (1978)
 A Method for Obtaining Digital Signatures and
 Public-Key Cryptosystems. Communications of the
 ACM, 11(4), pp. 331-356, 1979.
- Smid, M.E.(1981)
 Integrating the Data Encryption Standard into
 Computer Networks.
IEEE, Vol. COM-29, No. 6, pp. 762-772, June 1981.
- Tomkins, F.G. and Rice, R. (1985)
 Integrating Security Activities into the Software
 Development Life Cycle and the Software Quality
 Assurance Process, in J.B. Grimson and H.-J. Kuegler
 (Eds.) Computer Security: IFIP Sec. '85; IFIP '85
 Elsevier Science Publishers B.V., Amsterdam
- Wood, M.B. (1982)
Introducing Computer Security
 The National Computing Centre Limited, 1982
- X9.17, (1985)
Financial Institution Key Management (wholesale).
 ANSI, 1985.

V. COMPUTER INTEGRATED MANUFACTURING

Paper presented in the Plenary Sessions:

CNMA - Putting Standards to Work - P 955 **1533**

Parallel Sessions

- | | | |
|----|---|-------------|
| 1. | CAD and CAM Tools | 1547 |
| 2. | Knowledge based Developments in CIM | 1619 |
| 3. | Robots - Developments in Manufacturing Systems | 1693 |
| 4. | Evaluation of CIM Systems | 1745 |
| 5. | Human Factors (Part II) | 1795 |

Project No. 955

CNMA - Putting Standards to Work

John Booth (ICL)

Bernard Girard (Peugeot)

INTRODUCTION

In June 1988 it is planned to hold the Enterprise Networking Event 88 International (ENE) in Baltimore. This will be a showcase for the results of the MAP (Manufacturing Automation Protocol) and TOP (Technical and Office Protocols) initiatives started by General Motors and Boeing in the USA. The Event is a major milestone in these programmes leading to truly multi-vendor Computer Integrated Manufacturing (CIM). In the USA, these programmes, combined with government initiatives to introduce OSI (Open Systems Interconnection) standards into procurement, have stimulated interest and investment amongst both users and vendors.

It was seen as vital for European industry that there should be a European programme to complement the other international initiatives in the field of manufacturing networks. This led to the introduction of the CNMA project, which is supported by the Commission of the European Communities under its ESPRIT programme.

To provide the necessary combination of skills, interests and resources, the CNMA consortium comprises the following eleven major European companies :

Users	British Aerospace (prime contractor)
	Aeritalia
	BMW
	Peugeot SA
Vendors	Bull SA
	GEC
	ICL
	Nixdorf
	Olivetti
	Siemens
System Engineers	TITN

The main sub-contractors are Fraunhofer IITB, who are responsible for the development and application of conformance testing tools, and ELF Aquitaine to represent users in the process industry.

PROJECT AIMS

CNMA aims to select, implement, validate, demonstrate and promote communications standards for manufacturing, within the framework of the International Standards Organisation (ISO) reference model for Open Systems Interconnection (OSI).

Selection

The selection of unambiguous "profiles" of existing and emerging communications standards is the necessary first step. CNMA has concentrated on the upper layers of the ISO OSI reference model and in particular on MMS (Manufacturing Message Specification) and FTAM (File Transfer, Access and Management). This work complements other activities leading to version 3.0 of MAP and TOP specifications. The CNMA profiles are documented in an Implementation Guide.

Implementation

CNMA is implementing these profiles on CNC and PLC controllers and on mini-computers, on three types of Local Area Network interconnected by Routers:

- IEEE 802.3 CSMA/CD Baseband,
- IEEE 802.4 Token Bus Broadband and
- IEEE 802.4 Token Bus Carrierband.

Validation

To verify that these implementations conform to the standards, conformance testing tools are being developed and used to test the implementations.

It is intended that the conformance tools developed by CNMA should become part of the accepted conformance suite for MAP and TOP at ENE. This aspect of CNMA has grown in importance and an extension to the project has been proposed to provide a range of conformance tests for the event.

Demonstration

The first stage in the CNMA project culminated in a demonstration at the Hanover Fair in April 1987. The demonstration involved the operation of a typical manufacturing cell controlled by CNC and PLC controllers and computers from all five of the original vendors. It was the

first public demonstration of the use of MMS.

Live production facilities in BAe, BMW and Aeritalia will also be used to demonstrate CNMA communications software. CNMA intends to participate in ENE using a Wide Area Network connection so that manufacturing activity at BAe can be controlled remotely from systems at the event.

Promotion

Promotion of these profiles to encourage their widespread acceptance is being achieved via liaison with standards bodies, by publicising implementation guides, by release of conformance test tools and procedures to test centres and by display of project results through seminars, conferences and visits to pilot facilities.

THE EUROPEAN PERSPECTIVE

Factory automation has in the past been hindered by the high cost of intelligent controllers. As the cost of controllers has declined there has grown up a number of "islands of automation" each conforming to a proprietary set of communications standards. Total integration of a multi-vendor facility becomes prohibitively expensive. To avoid this penalty, the user must select a single vendor's protocols which restricts considerably the choice of devices.

What the user wants is a communications architecture supported by all vendors. Evolution towards Computer Integrated Manufacturing demands that common standards be adopted by both vendors and users, especially in the key areas of Industrial Local Area Networks and data management. This is what OSI, MAP and TOP aim to achieve.

MAP and TOP have done much to further the idea of Open System Interconnection. The success of these programmes has made the target of open, multi-vendor systems seem achievable. However this target has not yet been realised. CNMA is intended to lend further weight, especially in Europe, to the momentum which has been established by MAP. It must be stressed that CNMA is aiming for the same final profile of communications standards as MAP.

Adoption of MAP by European vendors is still far from complete. The European MAP Users Group (EMUG) was founded to promote MAP in Europe. CNMA has the same general

objectives as EMUG, but through the significant European Commission involvement, and the large commitment made by its partner companies to the programme, it is achieving some things which EMUG cannot: for example, implementation of the new emerging standards by several vendors.

CNMA also represents European users' needs. MAP opted initially only for broadband technology with token bus access but a recent study in Europe showed that 98% of LAN installations use baseband technology. CNMA includes baseband LAN as an option, choosing the same standard as that supported by TOP. This allows a user to choose a type based on cost, performance, installed base, maintainability, etc.

A limitation on MAP's success has been the wide range of options within the standards and the lack of a complete suite of conformance tests. If Open Systems are to be realised, we must be able to demonstrate interworking. Only then can a user confidently expect to buy two products, plug them into a cable, and have a working system. Perhaps the best test of the CNMA implementations is that they are being applied to real production machinery provided by the users. These "pilot facilities" also ensure that the standards are appropriate to industrial use and provide the opportunity to demonstrate the project's achievements.

To meet users' needs CNMA has selected unambiguous profiles of communications standards, incorporating both existing and emerging standards. This work complements other activities leading to version 3.0 of the MAP and TOP specifications. The profile is documented in an Implementation Guide to which all project communications software conforms. CNMA is encouraging the alignment of MAP with European profiles defined by SPAG (Standards Promotion and Application Group) and CEN/CENELEC.

To gain confidence in the profile definition and the effectiveness of conformance testing it is important that several implementations of the standards are tested and then shown to interwork. The Fraunhofer Institute, the project's independent testing organisation, obtained and developed the conformance testing tools and test procedures to verify the CNMA implementations in the first phase of the project. Detailed plans are being made for conformance testing in the next phases of the project.

Currently, in Europe as well as worldwide, there are various institutions actively developing tools for conformance testing of communication protocols in open or industrial computer networks. Each test tool is claimed to be "correct" and "complete" but in fact, uniform test functionality does not exist. Nor is there a consistent

interpretation of test results. This means that the test results obtained by test institutions may vary due to the use of different testing tools and different test result interpretations. This is the reason why the US MAP User Group recognises just one single set of test tools as being authentic - those developed by the Industrial Technology Institute (ITI), which are applicable to MAP 2.1.

The realisation of true industrial open systems will occur far sooner if, at an early stage, there is established and recognised worldwide one set of high quality, consistent test tools for manufacturing communications. Such tools would be targeted on the final industrial standard, MAP 3.0. This goal is the primary objective of a proposed new CNMA project and would be achieved by establishing a European consortium employing the relevant expertise and resources to develop, maintain, licence and support testing tools for manufacturing communication, built from existing technology bases.

The urgent requirement for test tools for the forthcoming MAP/TOP/COS Enterprise Networking Event provides a rare opportunity for Europe to establish and own a significant test suite by bringing together the necessary resources under the ESPRIT programme. Besides the existing CNMA partners, SPAG Services, the Fraunhofer Institute, The Networking Centre, ACERLI, Leeds University and the NCC will be involved. Agreements with the Corporation for Open Systems (COS) in the USA have been made to ensure the widest possible acceptance of these conformance tests. It is also intended that, after the event, the work will continue to integrate the COS and CNMA conformance tests.

CNMA COMMUNICATIONS PROFILES

One of the CNMA project's aims is to specify a communications system for computers and programmable devices and we are now going to look in more detail at the specification.

Communication in CNMA is based on the Open Systems Interconnection basic reference model (OSI/RM), which is defined by ISO in IS7498. It defines a framework for communications, i.e. the services to be provided to application processes, the breakdown of the communications software into 7 layers and the split of services between the layers. The model is also known as the "OSI seven layer model".

Each service requires protocols - specified interactions - between the two systems. Definitions of the services and protocols in each layer are provided in individual standards documents. However, a given service can be provided by a number of different protocol combinations in the lower layers. Hence, an additional document is required to identify the exact protocols used at each layer. Such a selection of protocols is known as a "profile".

The Phase 1 CNMA Implementation Guide defines the communications profiles chosen for the first phase of this project. It represents a considerable amount of work by the participating companies and its publication is one of the primary functions of the project.

When MAP 2.1 was specified protocols for layers 6 and 7 were not stable and so interim solutions were included. The main purpose of CNMA is to focus on layers 6 and 7, an area of particular importance in the definition of MAP 3.0.

For layers 1 and 2, CNMA provides a choice of three types of Local Area Network (LAN): IEEE 802.3 CSMA/CD Baseband and IEEE 802.4 Token Bus Broadband and Carrierband.

In CNMA Phase 1, layers 3 to 5 were defined to conform with MAP 2.1. These layers are considered to be stable and remain largely the same when evolving to MAP 3.0, other than the addition of recently defined Network Management functions.

For layer 6 (the presentation layer), which was absent in MAP 2.1, CNMA specified the use of a kernel subset, though it was not used for Hanover.

Layer 7 protocols were covered by two chapters. The first defines the Common Application Service Elements (CASE), now known as ACSE, which uses the latest ISO draft international standard protocol to provide association control.

The second chapter for layer 7 protocols defines the Manufacturing Message Specification - MMS, which is a protocol for passing messages between computers, numerical controllers and programmable logic controllers. This has been the greatest area of activity in the project. In Phase 1, CNMA specified Draft 5 of EIA Project 1393A.

It is not necessary or desirable for vendors to implement the complete MMS service, so subsets have been selected to provide the functionality required for the first demonstrations. They provide the ability to:

- up and down-load programs to NCs and PLCs from computers
- dynamically down-load programs to NCs (whilst they are controlling machining)
- report change of status from NCs and PLCs
- start and stop program execution in NCs and PLCs
- request status information from NCs and PLCs
- Read and Write variable data to NCs and PLCs
- transfer files between computers.

In all these services, the NC or PLC interacts with a computer, except for the case of Read and Write: Read and Write can take place between any pair of the three types of device.

The CNMA profile is not final but will evolve as the upper layer standards mature. In Phase 2 of the CNMA project most effort will be devoted to upgrading MMS, and incorporating FTAM (File Transfer, Access and Management), Directory Services and Network Management.

CONFORMANCE TESTING

For testing MMS and CASE, Fraunhofer developed a test system on a multi-computer system programmed in 'C' and running under UNIX System V.2. The reason for using such an environment is that the test tools developed are then highly portable, to permit future delivery to vendor or user sites, and to other test institutions and developers. This was the first such tool developed for an application protocol which will become the main feature of MAP 3.0.

For testing the lower layers of CNMA implementations for the Hanover Fair demonstration, Fraunhofer obtained the recognised MAP 2.1 test-bed from the ITI in Michigan. These test tools were initially run on a VAX 11/750. It is worth noting that the ITI has subsequently migrated its MAP 2.1 test system to a SUN workstation using the common UNIX operating system.

During the pre-staging for the Hanover Fair, the test tools were used to great effect, testing the vendors' new implementations. The outstanding success of the demonstration, achieved by reliable interworking of the separate devices, is evidence of the quality of the test tools developed so far.

USERS' NEEDS

One of the major difficulties for industrial networks arises from the multiplicity and severity of problems encountered by users.

Although many LAN studies have been made, very few of them concentrate on users' needs. This is why we have conducted a detailed analysis of the subject within the context of the CNMA contract. The report notes that high reliability is a major concern for most applications, in both batch manufacturing and the process control industry. This implies the use of dependable techniques for both hardware, software and especially for protocols. Other requirements include the need for low cost installations, high integrity installations that allow for redundancy, and installations which permit reconfiguration as factory layouts change.

The final report which consists of 18 chapters, examines a vast array of problems including training requirements, costs, time and reliability constraints etc.

It brings into question a certain number of views, such as broadband MAP being a unique solution to users needs for example. This 200 page study which also touches on the needs of process control forms an easily readable report and can be obtained from the Commission.

HANOVER FAIR

After agreeing on the functional profiles of the standards to be used, the different vendors developed hardware and software products in order to connect their equipment to 802.3 10 mbits/sec networks and to broadband or carrierband 802.4 networks. The profile definitions are contained in the CNMA Implementation Guide which is also available from the Commission.

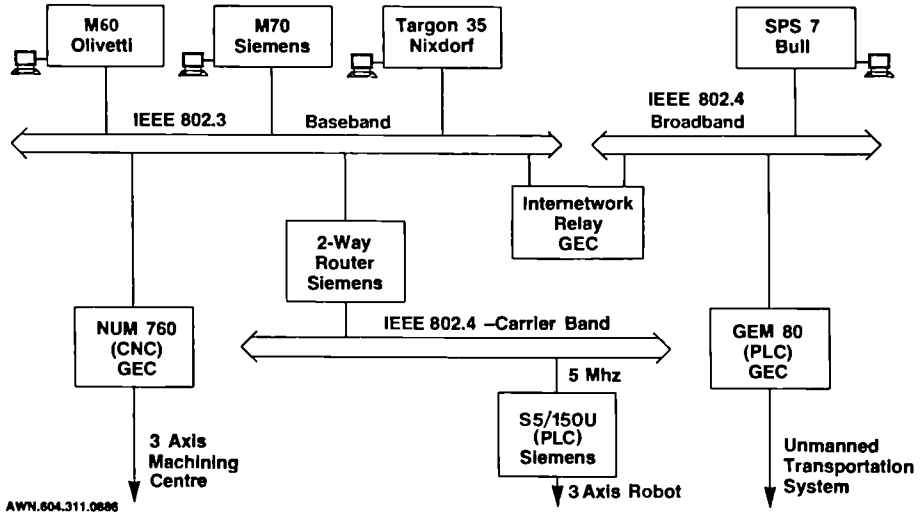
The first demonstration of these developments took place at the Hanover Fair (West Germany) from the 1st to the 8th of April 1987, as part of the Intermatic exhibition in hall 18.

During the 3 months preceding the Hanover Fair an integration centre at Wolverhampton (England) was used as a "pre-staging area" in preparation for the demonstration.

The different vendors' communication software (principally layers 1 to 5 and CASE/MMS) were tested by conformance test

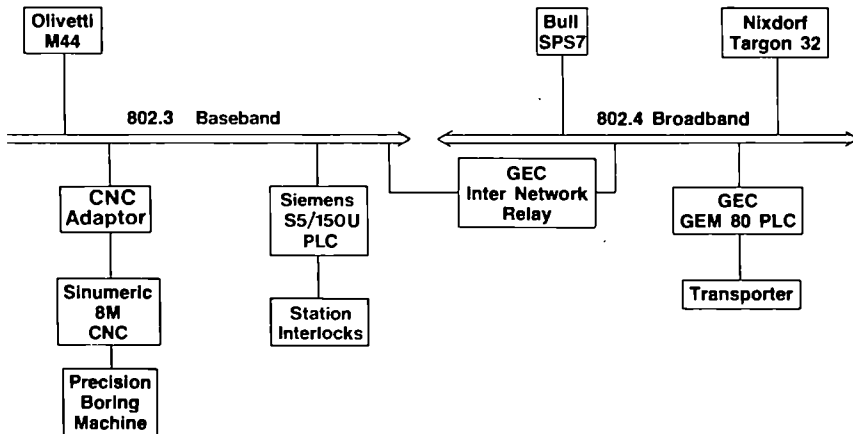
**BRITISH
AEROSPACE**

CNMA Hannover Cell Control Architecture



**BRITISH
EROSPACE**

A320 Cell - Control Architecture



equipment and software developed by the Fraunhofer Institute (IITB).

From early 1987, interoperability tests between the different vendors commenced, first using dummy messages and subsequently using application software.

Some general views of the Hanover Fair stand where the demonstration was run are shown in the illustrations.

The system architecture is shown with the three types of local area network connecting each vendor's hardware and the automated machining cell under control.

The cell consisted of a machining centre with its CNC and a robot and an automatic guided vehicle controlled by a programmable logic controller. The set of minicomputers and the application software provide the cell management and the operator interface.

This very successful demonstration was the first concrete manifestation of work for the Esprit 955 contract, and a world first in the use of MMS, a major component of the application layer that will be used for MAP 3.0.

PILOT FACILITIES

The CNMA project provides for a number of pilot facilities located at BAe, BMW and Aeritalia. The first pilots use manufacturers' software developed for the Hanover demonstration while the later pilots will incorporate changes to the standards to meet the requirements of ENE and MAP 3.0

Application software for the pilots is developed by BAe, BMW, Aeritalia, ICL and TITN.

BAe PILOT FACILITY AT SAMLESBURY

The pilot facility at Samlesbury will be used to machine parts for the A320 Airbus. This is essentially a Precision Boring Facility used to drill, bore and ream a large array of parts for A320 wing leading edges.

The workshop is divided into several areas (see diagram):

- a work preparation facility
- a cutting tool preparation and calibration facility
- a boring facility
- a bush installation facility
- a part conveyor system
- the machining facility itself. Machining is performed by a precision boring machine (KTM-FM200) controlled by a Siemens Sinumerik 8M CNC numerical controller.

Quality control is ensured by an on-machine positional probing facility to maintain machining precision.

The general architecture of the local industrial networks (see diagram) shows two networks (802.3 Ethernet and 802.4 Broadband) linked by a GEC inter-network relay. In the inter-networking example the structure has been deliberately made more complex than was strictly necessary.

The general philosophy is the same as for Hanover although here there is no carrierband and, for reasons of cost, the numerical controller is not directly interfaced with CNMA. Several approaches were examined before final selection of an Ethernet link between an ICL System 25 and a MicroVAX controlling the Sinumerik 8M.

The Bull SPS-7 and Nixdorf computers on 802.4 networks have functions similar in type to those used at Hanover although more extensive.

The Olivetti M44 mini-computer offers a gateway to the IBM world (SNA).

The Samlesbury pilot facility will be linked by a WAN (Wide Area Network) to the Aeritalia CNMA pilot facility and to the Enterprise Networking Event that will be held at Baltimore (USA) in June 1988.

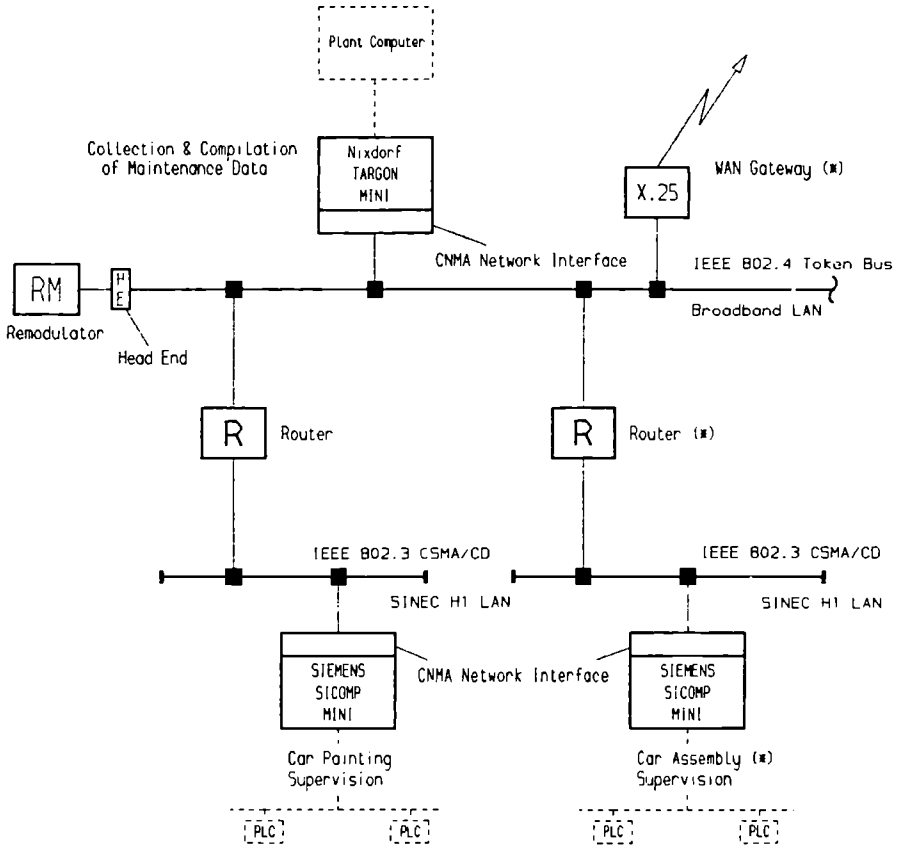
The pilot facility will be open to the general public for six months after start up.

BMW PILOT FACILITY AT REGENSBURG

As a result of increasing workload at BMW's factories 1 and 2 (Munich and Dingolfing), Bavarian Motor Works launched a third car factory in Regensburg, 140 km north-east of Munich, in November 1986. The BMW 3-Series and the BMW 325i Convertible cars are built there, mainly for export to the overseas markets.

The erection of a brand new plant eased the implementation of the latest production technology, e.g. JIT (Just-In-Time), and the build-up of a communications infrastructure with future demands in mind. The plant is covered by a broadband network, comprising 30 kms of coaxial cable and 600 connection points, mostly 4-pin taps.

Pilot 2 Facility at BMW's Plant in Regensburg, Bavaria



(*) Indicates optional implementations, which will only be provided if resources and timescales allow.
The dotted lines (---) indicate equipment and protocols which are outside the CNMA architecture.

The diagram below shows the general architecture of BMW's pilot. The key elements are a Nixdorf TARGON fault-tolerant mini which is connected to an IEEE 802.4 Token Bus LAN, and two Siemens SICOMP minis, each of them linked to an individual IEEE 802.3 baseband LAN. Connection between the broadband LAN and the baseband LANs is established via routers from Bull and/or GEC.

As pilot application BMW chose the collection and compilation of maintenance and repair data from the factory floor to facilitate continuous operation of the production machinery.

Approximately 200 PLCs continuously monitor devices connected to them and relay error condition information to the Siemens SICOMP minis. These minis supervise car painting and car assembly, respectively.

An initial analysis and pre-processing of this information is performed by the Siemens minis, in order to decide which data must be forwarded immediately to the Nixdorf TARGON mini. The remaining data will be stored on disc to be dispatched as a file to the Nixdorf mini at shift's end.

The transmission of the (short) messages as well as the transference of the (mega byte) files is accomplished by means of MMS within the CNMA protocol framework.

The BMW pilot will be fully operational by February 1988 to demonstrate the first CNMA/MMS implementation on a factory floor.

AERITALIA PILOT FACILITY AT TORINO

Aeritalia's CNMA pilot facility concerns production and control of cableforms (looms or harnesses) destined for military aircraft (combat Aircraft group).

The CNMA pilot facility fits into an overall system providing

- design and engineering used to prepare harness drawings including electrical properties
- industrialisation which provides tooling and testing data to the manufacturing area.
- manufacturing, the last subsystem, covering both manufacturing and assembly.

The pilot facility will be installed in the cable preparation area.

The purely CNMA part comprises:

- a TAB MOD MA 200 DD cutting and stamping machine which at the present time is controlled manually from a front panel. Cable data (type, length, etc.) is entered manually from typed documents.
- an Olivetti PLC which controls the TAB machine.
- an Olivetti L1 mini-computer supplying the information required by the PLC (loom manufacturing parameters). It also serves as a support for a connection with the BAe facility over an FTAM X.25 link.
- A Bull SPS7 whose main role is to create and update a technical information database using data from a DEC VAX 8250. It also receives production orders entered by operator and optimises cutting and marking operations effected by the TAB machine.

This set of equipment is connected to an 802.3 10 base 5 network which also includes (see diagram) the VAX 8250 and three PDP 11/23 systems performing final test operations, plus a PDP 11/84 performing automatic creation of test programs that are remotely loaded into the PDP 11/23 systems.

Starting from CAD on a Computer Vision CGP 200X over a 2780 link to an IBM 3032, itself connected in SNA mode to the VAX 8250, the whole data stream, right through to final control, is thus automated.

This pilot facility will be operational in October 1988.

Project No. 322

CAD data transfer
The goals and achievements of Project 322,
CAD Interfaces (CAD*I)

E. G. Schlechtendahl, U. Gengenbach

Kernforschungszentrum Karlsruhe GmbH
 P.O.B. 3640
 D-7500 Karlsruhe 1, Federal Republic of Germany

Project 322 (CAD Interfaces, in short: CAD*I) is concerned with the development of methods and software associated with the various interfaces of CAD/CAM systems. Three out of the eight working groups of the project deal with the main issue of CAD modeling: geometric models represented by curves, surfaces, and solids, and with the transfer of such data between CAD systems and from CAD to other CIM areas. One working group is concerned with interfacing CAD systems to data base management systems and to computer networks; a further working group develops new techniques for a powerful AI based human interface. Finite element modeling, analysis, and dynamic experiments, whose results are to be fed back into design evaluation are the subject of three other working groups.

In this paper, besides giving a project overview, emphasis will be on CAD data transfer. The following topics will be addressed in some detail:

- The state of the art prior to initiation of the CAD*I project.
- The methodology developed by CAD*I for rigorous specification of CAD data structures in syntactical and semantical respects.
- The strategy and status of processor development.
- The results obtained so far.
- The impact of the CAD*I project on the international standard for product model data (STEP).

1. PROJECT OVERVIEW

The objective of the ESPRIT Project No. 322 is to develop a set of standard CAD interfaces and to contribute to national and international standardisation efforts. The project is planned for five years starting from November 1984. The project team consists of twelve partners:

- Bayerische Motorenwerke AG (Germany)
- CISIGRAPH (France)
- Cranfield Institute of Technology (United Kingdom)
- Danmarks Tekniske Højskole (Denmark)
- ERDISA (Spain)
- Gesellschaft für Strukturanalyse (Germany)
- Katholieke Universiteit Leuven (Belgium)
- Kernforschungszentrum Karlsruhe GmbH (Germany), prime contractor
- Leuven Measurement and Systems (Belgium)
- NEH Consulting Engineers APS (Denmark)
- Rutherford Appleton Laboratory (United Kingdom)
- Universität Karlsruhe (Germany)

The research and development efforts are organised in eight working groups:

- CAD data exchange interfaces for:
 - wireframe models (working group 1)
 - solid models (working group 2)
 - surface models (working group 3)
 - network access (working group 4/network)
- CAD database interface (working group 4/database)
- advanced modeling interface (working group 5)
- interfaces to Finite Element systems and experimental analysis systems
 - FEM model description (working group 6)
 - FEM model optimisation (working group 7)
 - experimental dynamic model description (working group 8)

The work in these working groups has been reported previously [1], [2], [3], [4]. In this paper we will emphasize the work in the area of CAD data transfer, especially with respect to the impact which this work has already had on the international standardisation.

2. THE STATE OF THE ART IN SOLID MODEL TRANSFER

In the early days of computer aided design emphasis was mainly on drafting (two-dimensional representations) and an approximate representation of three-dimensional objects with wireframe models. In certain areas, mainly in the automotive industry, in aircraft and spacecraft design, and for shipbuilding, modeling techniques for sculptured surfaces grew up rapidly. More recently, the need for proper representation of solid models becomes more and more evident in particular in the mechanical products industry and in plant layout.

The history of CAD data exchange parallels this development. Originally, transfer of drawings was satisfactory. It is not surprising that the still most widely used standard was called IGES = Initial Graphics Exchange Specification, with emphasis on the graphics aspect. Two-dimensional and three-dimensional line geometry, and to some extent also non-geometrical information can be transferred with IGES, if not satisfactorily but at least suitably for most practical applications. Another standard for transfer of similar information is SET. Transfer of surface information has also become feasible. It is now covered in IGES [6], SET [8], and the VDA-FS [7] standard. Transfer of solid model data, however, is still a matter of research and development.

It has become evident in the past years that there is a need for an integrated mechanism for transfer of all product information. Especially, all types of geometric data have to be covered by a single transfer specification. The international standard STEP is to provide this facility. The CAD*I project has first concentrated on solid model information because this type of data poses the most difficult problems in terms of data structuring and efficiency requirements. A specification devoted to solid model transfer (version 2.1) was developed and used as a basis for processor development [15]. A new specification which integrates curve, surface, and solid models has since been prepared and will be used for subsequent CAD data exchange. We will report here on the results of data exchange tests based on specification 2.1 and on briefly summarize the contents of specification 3.2.

2.1 SOLID MODEL REPRESENTATIONS

Before going into the details of solid model exchange a short look at the solid model representations relevant in that context seems sensible. According to Requicha [5] there are six families of unambiguous representations of solid models. Among those only three have come to a larger use in commercial solid modeling systems and are thus candidates for solid model exchange.

- Boundary representation (B-rep)

Intuitively a boundary representation describes a solid object in terms of its enclosing surfaces. The B-rep structure is composed of a topological structure and associated geometry. The topological information consists of a graph of vertices and edges embedded in a surface. The sequence of edges that bounds a region of the surface, termed face, is called loop. The POLYHEDRON structure is a special case of the B-rep structure. It represents objects bounded by planes.

- Constructive Solid Geometry

A CSG model represents the object by a tree whose nodes are boolean set operators and whose leaves are either bounded primitives or halfspaces. The boolean operators are union, difference and intersection. The set of primitives consists e.g. of block, cylinder and cone. Halfspaces are defined by an infinite open surface that divides space into two distinct regions, solid and void.

- Sweeping

The set of points defining a sweep object can be represented as the Cartesian product of an area set and a trajectory set. Intuitively the solid can be regarded as being created by sweeping the area along the trajectory.

The solid of linear extrusion is a special case of the general sweep. It is defined by a planar area and a vector as trajectory.

Some systems allow for sweep objects and B-rep objects as leaves of the boolean tree. Others store an object by both an unevaluated representation, the CSG tree, and an evaluated representation, the B-rep. This variety of representations defines the requirements on an exchange specification.

2.2 THE EVOLUTION OF CONCEPTS FOR SOLID MODEL EXCHANGE

The development of solid model interfaces is strongly influenced by CAM-I (Computer Aided Manufacturing - International Inc.) (see Figure 1). In the late 1970s with most solid modelers still in research laboratories and universities CAM-I already identified the need for solid model exchange. CAM-I gave a contract to McDonnell Douglas Automation (McAuto) to prepare a file specification for solid models. That effort was completed in 1979. Part of that specification was cast in the IGES format and published as "Section 5 - Basic shape description" of ANSI Y 14.26 M (IGES 1.0) [6].

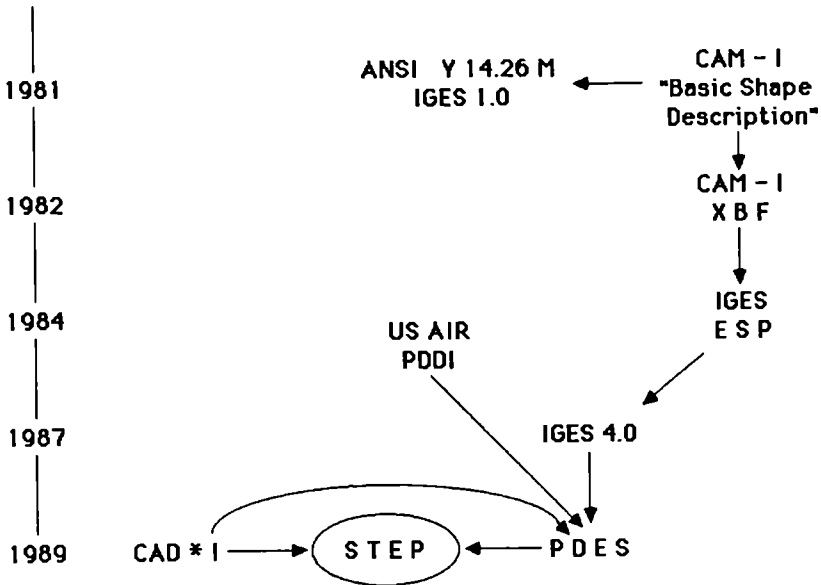


Figure 1. The development of solid model interfaces

2.2.1 IGES Section 5 - Basic shape description

Due to the fact that the basic shape description was a result of the McAuto work, section 5 was completely independent from the first four sections of IGES 1.0. It contained its own geometry definition for B-rep solid models though most of the basic geometry, was already defined in the first four chapters of ANSI Y 14.26 M. The only link was the use of the same basic file format.

The basic shape description relied on a relational table format which was represented in the IGES format by the Relation Entity (entity number 500) and the Domain Structure Entity (entity number 502). Any structure for basic shape description is defined by using either of those two entities.

A model was composed of three classes of entities:

- Geometric entities (Curves, Surfaces)
- Topological entities (Vertex, Edge, Loop etc.)
- Miscellaneous entities (Group)

There are no hints so far that the basic shape description has ever been used for solid model transfer.

2.2.2 Experimental boundary file (XBF)

In parallel with the work at McAuto, CAM-I launched a project to develop a neutral subroutine interface to a modeler. The "Applications Interface Specification" (AIS) was developed by Ian Braid of Shape Data Ltd. and finished in 1980. It consisted of about 150 subroutine calls, which allow to access

both CSG and B-rep modelers. The AIS has since been restructured and enhanced a number of times. The latest version was released in January 1986.

With the experience of those two specifications CAM-I decided to produce a new specification the "Experimental Boundary File" (XBF). The first version of XBF was published in 1981 and a revised version in 1982 [10]. XBF uses the IGES format as physical file format. The topological and CSG entities are defined as new IGES associativities (entity number 402 with form numbers 100 - 110) and the geometric entities as new IGES entities (entity numbers 700 - 744).

XBF supports both B-rep and CSG models. The B-rep model is based on the topological structure consisting of Vertex, Edge, Path, Face, Shell and Object with associated geometry. That structure allows for backpointers to entities on the next higher level in the topological structure e.g. back from the Path to the Face. A Path is an ordered list of Edges. In case that the Path is closed and bounds a region of a surface it corresponds to the above explained Loop. However the specification allows for a generalization of that structure by introducing dummy faces with no associated geometry. This is a means to represent special cases of solid models such as sheet objects with associated thickness or graphs of piping systems with associated cross section.

XBF provides for CSG models the boolean operations Union, Intersection and Complement and a set of CSG primitives:

- Cuboid
- Circular and elliptical cylinder
- Circular and elliptical cone
- Sphere
- Ellipsoid
- Pyramid
- Prism
- Torus
- Halfspace

Similarly to the "Basic Shape Description" XBF contains its own set of geometric entities ranging from analytic to parametric geometry. The objective was to avoid one of the weaknesses of IGES by allowing for symbolic referencing instead of comparing numeric values. For that reason the geometry is defined as far as possible by some basic building blocks such as Point, Direction and Local Coordinate System.

An implementation of XBF was done at Lucas Industries Ltd. to prove its viability and the viability of the AIS. Cycle tests have been performed with e.g. the MBB test part (see Figure 12). Already those first tests showed that the efficiency of the translation of an XBF file was rather bad. The post-processor spent most of the time searching through the pointer structure in the PD and DE sections of the file. In addition the file format was rather space consuming because it was initially tailored to fulfill the needs of IGES. The solid model data used e.g. only six of the 20 fields in a DE entry [11].

2.2.3 The IGES Experimental Solids Proposal (ESP)

In 1983 the IGES community felt the need to enhance the IGES specification with the capabilities to transfer solid models. The ESP was specified on the basis of XBF. It supports B-rep and CSG models [12].

The ESP lost the generality of the topological structure defined in the XBF i.e. it does not provide for sheet and graph objects. As a new feature the "Face-set" and the "Edge-set" entities were added to define collections of faces and edges and attach additional information to them.

The curves and surfaces geometry is except for some minor differences the same as in XBF.

On the CSG level some primitives of lower significance such as elliptical cone were removed and some new ones such as Solid of Linear Extrusion were added.

Those entities were embedded in the IGES physical file format as separate entities and no longer as associativities as in XBF.

In 1984 the ESP was frozen for a testing phase of two years. After that period it turned out that no testing of the B-rep part had been done. The CSG part has been successfully tested between PADL-2 at Ford Motor Company, GMSolid at General Motors and TRUCE at General Electric.

2.2.4 IGES 4.0

After that successful testing the CSG part of ESP was used for the enhancement of IGES 4.0 [13]. The following CSG entities are included in IGES 4.0:

- Primitives:
 - Block
 - Right angular wedge
 - Right angular cylinder
 - Right circular cone frustum
 - Sphere
 - Torus
 - Solid of Revolution
 - Solid of Linear Extrusion
 - Ellipsoid
- Boolean Tree
- Solid Instance
- Solid Assembly

2.2.5 The Product Data Definition Interface (PDDI)

The PDDI effort initiated by the US Air Force is intended to provide a complete computer based product definition interface between design and manufacture. Its approach is somewhat similar to that of the XBF and the AIS in that way that it supports both a static file and a "working form" as access medium for application programs. PDDI comprises entities for geometry, topology, tolerances, part features and part control information. As a result of a military project it is not available for commercial use; however, it has occasionally been demonstrated in public [14].

2.2.6 SET solids proposal

In 1987, Aerospatiale will publish a proposal for transfer of solid models that is to be included in the French SET standard. The proposal covers both CSG and B-rep models.

2.3 TOWARDS A NEW STANDARD: STEP

From this short summary of previous development it becomes evident that the US activities have so far dominated the development. Since the CAD*I project was started the situation has changed in the following respect:

- on the international level ISO/TC184/SC4 has initiated the project of developing a single world-wide Standard for Product Data (STEP). The USA, UK, Japan, France, and Germany are most active in this project. All countries (except France which continues to further develop SET) have channelled their activities towards converging on a single standard.
- The US delegation to ISO/TC184/SC4/WG1 has been appointed to lead this enterprise on the basis of their national efforts called PDES (Product Data Exchange Standard). The US National Bureau of Standards has announced that IGES will not be developed beyond the version 4 but that all resources will then be concentrated on STEP/PDES.
- The CAD*I project has become the best-known and most successful contributor to the foundations of STEP development. The CAD*I proposals on representing curves, surfaces, and solids were presented and successfully defended on the ISO level by CAD*I members who were at the same time members of their national (British and German) delegations to ISO. Furthermore, the specification techniques developed by CAD*I and the elements of the neutral file language for the physical file format have had a significant impact on what is the present STEP proposal.

3. THE CAD*I SPECIFICATION FOR CAD DATA

Originally the CAD*I specification work for CAD data concentrated on solid modeling [15]. In this first specification curve and surface information was included only to the extent that was necessary to prove the feasibility of the whole approach. This meant that for boundary representations of solid models no sculptured surfaces were implemented; similarly, for CSG modeling the contours of sweep primitives were restricted to connected segments of lines and circles. This situation has now changed and the new specification for CAD data, version 3.2, integrates all geometric modeling types. The following is a brief summary of the information that is included in the new specification [16].

3.1 THE GENERAL APPROACH

One of the lessons learned from the history of CAD data exchange is that much more emphasis must be put on **what to transfer** rather on **how to transfer**. The semantics aspects dominate over the syntactical aspects. Hence, the specification of data transfer mechanisms concentrates on the specification

of CAD data structures first. Three levels of semantics have been identified for the CAD data structures:

1. the data structuring level,
2. the reference model level,
3. the user level.

In addition to these there is a need to define semantics for the neutral file format on

4. the neutral file format level (the physical layer).

3.1.1 The specification language

A formal specification language HDSL (high level data specification language) was developed for specifying CAD data structures (what to transfer). This language is based on concepts of Pascal data structures, but introduces new features such as the scope concept. The scope concept introduces the block structure facility well-known in programming languages into data structure specifications. Complex information structures may be treated as single "black boxes" from the outside while still maintaining the possibility to enter into the details. HDSL has evolved from an early PDES proposal for such a language (DSL) [17].

3.1.2 Data structuring level

The general approach on this level is borrowed from the concepts of abstract data types: semantics are defined by the result of certain functions operating on the data structure. The operations considered on this level are the elementary functions of data structuring:

- enter (the scope of) an entity,
- leave (the scope of) an entity,
- create an entity or property,
- delete an entity or property,
- identify an entity (e.g. by picking its graphical representation on a display screen),
- invoke a function while passing entities and properties as parameters to that function.

The terms entity, property, reference, and scope are informally introduced below (see "Basic elements of the specification language HDSL"). The above functions allow to define these inherent features of all reference models which can be specified with the HDSL much more precisely and lend themselves to a proper formal specification. For instance, the behaviour of an ENTITY (which can exist in a certain scope without depending on other information) and a PROPERTY (which always depends on the existence of other information) can be precisely defined.

3.1.3 The reference model level

On this level we deal with the semantics of the reference model as defined in the specification. As a principle, the semantics on this level are described by descriptive text and supported by graphical representations.

As most of the reference model concerns geometrical information, parametric mathematical formula are used throughout the specification.

3.1.4 The user level

During actual use of a CAD system the model will have a specific meaning to the user. This meaning will depend on the particular application. The semantics on this level is left open in the specification.

3.2 BASIC ELEMENTS OF THE SPECIFICATION LANGUAGE HDSL

Some basic building blocks are used for the definition of the reference schema that are application independent

Entity	Entities are collections of data attributes. Entity types are defined as an ordered sequence of such attributes. Attribute types may be elementary such as integer, real, string etc.. or may be of some composite attribute type that will be declared explicitly.
References	Entities have a name and may thus be referenced. There exist two types of references: <ul style="list-style-type: none"> • many to one references termed "REFERENCE" • one to one references termed "REF_ONLY" One to many references are represented by a list of the above references.
Property	Properties are a means of attaching additional attributes to an entity via reference. This mechanism is used to attach application dependent data to an entity e.g. a material property.
Scope	The Scope mechanism of the CAD*I specification allows to build block structures of entities in a similar way as some programming languages such as Pascal or Algol. The idea is that entities should be able to contain other entities, such that the interior structure of the enclosing entity is hidden from the outside. That black box concept guarantees that references are not allowed to entities enclosed in the scope of another entity.

The concept of Scope is a means to ensure not only the correctness of the transferred data but the correctness of the operational model as well. The intent is to guarantee that with respect to basic operations such as create, delete identify etc., the transferred model behaves in the receiving system in the same way as in the native system.

The data structures of the schema are described formally by the "High Level Data Specification Language" (HDSL). The semantics are defined by the sequence and the naming of the attributes in HDSL and informally by verbal descriptions.

3.3 GENERAL DATA BASE STRUCTURE

The WORLD entity represents the complete content of a data base. Its major constituents are ASSEMBLY entities which may be nested recursively to indicate the assembly structure of a product. The COMPONENT entity is the elementary object in such a structure and it has geometry associated. A compo-

nent may have different geometrical representations according to different aspects. Two library mechanisms are supported: the PART_LIBRARY for standard parts which may be referenced from many data bases, the ROUTINE_LIBRARY as a library for modeling functions which the CAD operator may invoke.

3.4 REFERENCING MECHANISMS

Besides internal referencing (internal to one data base) the CAD*I reference model supports referencing of parts in a library. Furthermore, the neutral file may contain references to entities which already reside in the receiving data base. Such references are resolved by the post-processor.

3.5 GEOMETRIC MODEL ENTITIES

The class of GEOMETRIC_MODEL comprises WIREFRAME_MODEL, SURFACE_MODEL, and SOLID_MODEL.

3.5.1 Points and curves

The reference model supports a class of elementary curves which consists of LINE (infinite), LINE_SEGMENT (between two points), CIRCLE, ELLIPSE, HYPERBOLA, PARABOLA, POLYGON, and a general B_SPLINE_CURVE. A polynomial curve representation is also specified. All curves are defined in parametric form. Based on these the class of derived curve entities contains TRIMMED_CURVE and COMPOSITE_CURVE entities. Trimmed curves are finite sections of other curves (including composite curves); composite curves are connected series of trimmed curves. The OFFSET_CURVE allows to define a curve parallel to another curve.

3.5.2 Surfaces

The class of elementary surfaces comprises PLANAR_SURFACE, SPHERICAL_SURFACE, CONICAL_SURFACE, CYLINDRICAL_SURFACE, TOROIDAL_SURFACE, B_SPLINE_SURFACE, SURFACE_OF_REVOLUTION, and SURFACE_OF_TRANSLATION. A polynomial surface representation is also specified. All surfaces are defined in parametric form. Surfaces may be trimmed in rectangular form in the (u-v) parameter space to give RECTANGULAR_TRIMMED_SURFACES which may be combined to larger patches as a RECTANGULAR_COMPOSITE_SURFACE. Surfaces may also be trimmed by irregular boundaries as a CURVE_BOUNDED_SURFACE. The OFFSET_SURFACE is a surface parallel to another surface.

3.5.3 Geometry on surfaces

The CURVE_ON_SURFACE, the POINT_ON_SURFACE, and the DIRECTION_ON_SURFACE represent geometry that is defined on a parametric surface. Most of the two-dimensional curve entity types have a correspondence in this class. These curves are defined in the (u,v)-space spanned by the two parameters of the surface to which they belong. From this space they are then mapped into the three-dimensional Cartesian space of the world coordinate system.

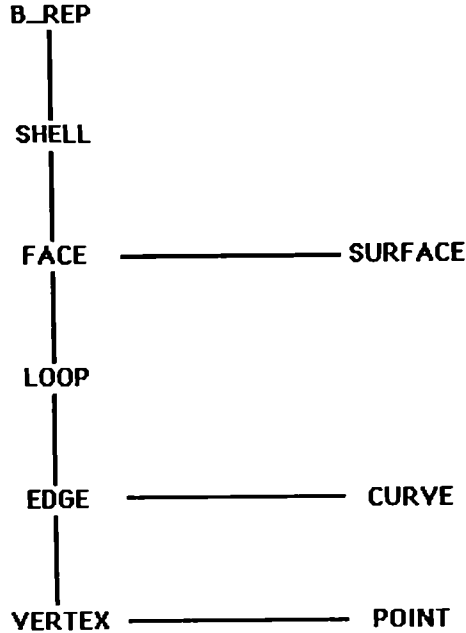
TOPOLOGYGEOMETRY

Figure 2. The B-rep data structure

The INTERSECTION_CURVE may have up to three representations: In three-dimensional space and as a curve in each of the parameter spaces of the two intersecting surfaces.

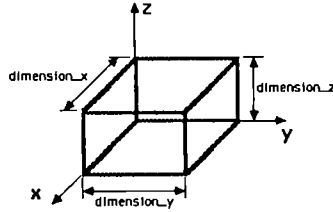
Derived curves on surfaces correspond to derived two-dimensional curves (TRIMMED_CURVE_ON_SURFACE, COMPOSITE_CURVE_ON_SURFACE).

3.5.4 Boundary representations

The standard boundary representation is based on the topological entities B_REP, SHELL, FACE, LOOP, EDGE_LOOP, EDGE, VERTEX_LOOP, VERTEX, and on the geometric entities associated with faces and edges FACE_SURFACE and EDGE_CURVE.

The POLYHEDRON is a special boundary representation with planar faces only. Its data structure is simpler than that of a general B_REP (POLY_HEDRON, POLY_SHELL, POLY_FACE, and POLY_LOOP).

A compound boundary representation has been introduced to allow for volumetric models with internal faces, possibly separating regions of different material properties.



```

HDSL description:
ENTI <BOX> = STRUCTURE
    dimension_x : ANY (REAL);
    dimension_y : ANY (REAL);
    dimension_z : ANY (REAL);
    placement   : ANY (PLACEMENT (D3));
END;

ATTR PLACEMENT = GENERIC (TYPE OF DIM)
    STRUCTURE
        rotation : ROTATION (TYPE);
        translation : POINT (TYPE);
    END;
    
```

```

Realisation on the file:
BOX (#7: +6.00000E+01, +6.00000E+01, +3.00000E+02, PLACEMENT
(:ROT_MATRIX (:DIRECTION (:+1.00000E+00, +0.00000E+00,
+0.00000E+00 :), DIRECTION (:+0.00000E+00, +1.00000E+00,
+0.00000E+00 :), DIRECTION (:+0.00000E+00, +0.00000E+00,
+1.00000E+00 :)), POINT_CONSTANT (:+7.00000E+01,
+7.00000E+01, +0.00000E+00 :));
    
```

Figure 3. Example of a BOX primitive in HDSL and on the neutral file: The HDSL description and the file format correspond to version 2.1 of the specification.

3.5.5 Constructive solid geometry

A CSG model (called CONSTRUCT) consists basically of a data structure that represents a boolean expression (union, intersection, and difference operators are supported) of PRIMITIVE entities.

The primitives supported by the specification are: PLANAR_HALFSPACE, BOX, REGULAR_PRISM, SOLID_CYLINDER, SOLID_SPHERE, TRUNCATED_CONE, TRUNCATED_PYRAMID, SOLID_TORUS, CONTOUR_ELEMENT, LINEAR_SWEEP, and ROTATIONAL_SWEEP.

The HYBRID_SOLID entity corresponds in the field of solids to what the REAL and INTEGER entities are in the field of arithmetics (see below "Parametric modeling"). A HYBRID_SOLID consists of a functional description of a solids model in the form of a CONSTRUCT (referenced by the attribute expression) and its evaluated value in the form references to a list of B_REP models (referenced by the attribute value).

3.6 GENERAL GROUPING MECHANISM

A general grouping mechanism is provided which allows to associate geometric entities to certain ASPECT entities via a GEOMETRY_ASSOCIATION.

3.7 PLACEMENT AND INSTANCING

Placing geometric objects in space is defined as a rotation followed by a translation. Any number of INSTANCES of a geometric object may thus be produced. Instances of instances are also supported. Various forms for defining a rotation are allowed (around an axis in space, around the global coordinate axes, or a rotation matrix).

3.8 TEST DATA ELEMENTS

The relations TEST_RELATION_FOR_D2_WIREFRAME, TEST_RELATION_FOR_D3_WIREFRAME, TEST_RELATION_FOR_SURFACE_MODEL, and TEST_RELATION_FOR_SOLID_MODEL express the fact that certain points on the neutral file have been produced in the sending system by intersecting the model with straight lines or planes which are also transferred on the file. The CAD system user in the receiving environment may use this information to test the accuracy of the transmitted geometry by repeating this intersection operation.

3.9 MISCELLANEOUS

The MATERIAL was included as a representative of arbitrary properties which may be attached to entities.

An escape mechanism was provided to allow for a standard way for representing non-standard information on the neutral file as RECORDs possibly subject to RECORD_TYPE restrictions.

For interfacing with non-CAD data bases a special mechanism is supported (DATA_BASE_BRIDGE, DATA_BASE_LINK).

3.10 PARAMETRIC MODELING

Different forms of parametric modeling are allowed. Entities of type INTEGER and REAL permit the use of variables and arithmetic expressions in the data structure instead of constant values. Thus functional dependencies of geometric shapes can be expressed. It is, e.g., possible to define a box with side length ratios 1:2:3 such that assignment of a value to a single real entity will change the size of that box consistently.

MACRO and ROUTINE facilities represent functions which the CAD user invoke during a session to generate new shapes.

3.11 LEVELS OF SCHEMA IMPLEMENTATION

It would be unrealistic to expect all CAD systems to support the full spectrum of capabilities covered by the CAD*I specification with their pre- and post-processors. On the other hand, CAD data exchange becomes impracticable if the various system support arbitrary subsets of the specification. Hence, a number of well-defined subsets have been specified in four different domains. The subsets are characterised by four sets of short strings called levels:

1. The geometric modeling capabilities (l_g)
2. The capabilities of defining assembly structures (l_a)
3. The capabilities for parametric models and macros (l_p)
4. The capabilities for references (l_r)

That subset information is contained in the neutral file header. The subset defined in the header of the neutral file (line 3: "...,'4','3','0','0',...." in Figure 3) specifies the following capabilities:

1. Geometric modeling capabilities $l_g = 4$: Hybrid models, both CSG- and POLYHEDRON models
2. Capabilities of defining assembly structures $l_a = 8$: Full assembly structure
3. Capabilities for parametric models and macros $l_p = 0$: No parametric models, no macros
4. Capabilities for references $l_r = 0$: No external references, no library references

4. TRANSFER OF SOLID MODELS - RESULTS OBTAINED SO FAR

4.1 THE IMPLEMENTATION OF THE CAD*I INTERFACE FOR SOLID MODEL EXCHANGE

It is one of the goals of the CAD*I project to test the specification by implementing processors for a number of commercial CAD systems. The systems involved in implementing specification version 2.1 are shown in Figure 4. After publication of version 3.2 of the specification these processors will be adapted to the new specification.

The general approach to processor development is characterized by the utilisation of common software (see Figure 5).

All pre-processors are structured in the following phases:

1. inspect the CAD system's data base,
2. build a data structure representation conforming to the CAD*I specification,
3. write the information to the neutral file.

The last one of these phases is implemented in a set of subroutines that is available for all pre-processor development.

All post-processors are structured in the following phases:

1. scan the neutral file,
2. parse the neutral file,
3. build a data representation conforming to the specification,
4. perform semantic checking,
5. translate the information into the representations of the particular CAD system and write it into the CAD system's data base.

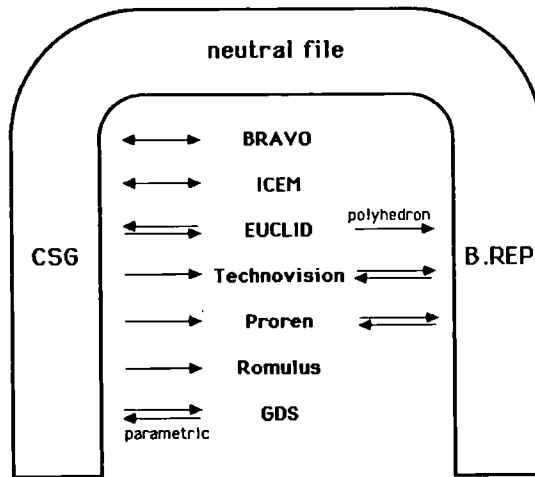


Figure 4. Systems involved in the processor development for CAD*I Solids

(according to specification 2.1) common scanner software was available for all post-processor development. For version 3.2 of the specification common scanner and parser software has been prepared [18]. It is planned also to provide further common software including semantic checking.

4.2 RESULTS OF CAD DATA EXCHANGE TESTS

More than a dozen test files have been set up to test special features of the CAD*I specification and the corresponding processors. The tests were very satisfactory and proved the appropriateness of the CAD*I approach. Some of the intersystem exchanges were in fact first ones of their kind in the history of CAD data exchange. This applies especially to the intersystem transfer of boundary representations for solid models. We will now give examples of some of the test parts together with their exchange history.

The test part "basic CSG" as shown in Figure 6 is for testing basic CSG capabilities. It combines few CSG primitives with all the boolean operations (intersection, union, and subtraction). It passed the cycle test on Bravo3.

The test part "Spinning Top" as shown in Figure 7 is also for testing basic CSG capabilities. This test part was produced on Euclid. Its neutral file representation is shown in Figure 8 which was successfully transmitted to Bravo3.

The test part "EU/robot" as shown in Figure 9 is a more complex CSG model. It was produced on Euclid, then transmitted to the Bravo3 system from where it was sent to Technovision. As Technovision is a B-rep system it was

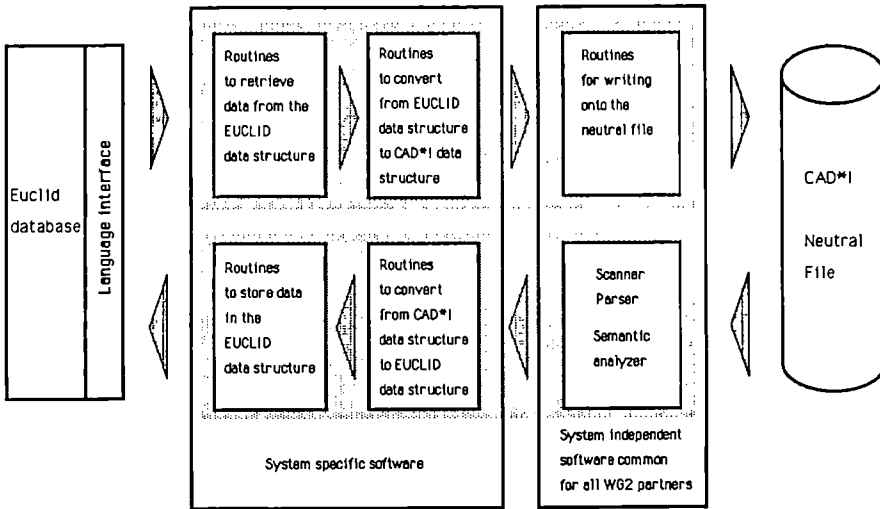


Figure 5. The Euclid processors: An example for the general structure of CAD*I processors

translated from a CSG representation into a B-rep representation at the end of this transfer.

The test part "Lego brick" as shown in Figure 10 is for testing basic B-rep capabilities. It passed the cycle test on Technovision.

The test parts "ANC 101" as shown in Figure 11 and "Gehauese" as shown in Figure 12 are well-known in the CAD community. They have been produced on Romulus and successfully transferred to Technovision via the CAD*I neutral file. The very first transfer of a B-rep model between unlike CAD systems was a transfer of one of the CAD*I B-rep test files from Proren to Technovision.

The major change that will take place when specification version 2.1 is replaced by version 3.2 will be that sculptured surfaces will be supported for B-rep models.

5. THE INFLUENCE ON INTERNATIONAL STANDARDISATION

The ISO/TC184/SC4/WG1 is concerned with the development of the future standard STEP (Standard for the Exchange of Product model data). It has been agreed among all ISO members that the American PDES project has the leading role in the development of STEP. However there are important contributions by other groups and projects. Among these, the CAD*I project turned out to become an important multinational European partner as its R&D work did not only complement the ISO specification task but provided also early experience from processor implementation and test of actual solid model exchange.

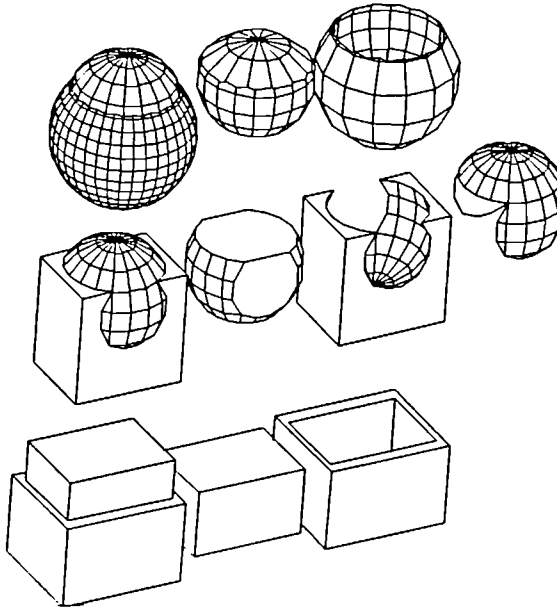


Figure 6. The test part for basic CSG capabilities

The members of the CAD*I project as representatives of their national standardization bodies contribute to each of the three layers of STEP.

5.1 SPECIFICATION TECHNIQUES FOR CAD DATA STRUCTURES

5.1.1 The data structure specification language

At present, the experience gained with the CAD*I specification language HDSL is used to influence the data specification language EXPRESS which is to be used for the STEP standard. Certain HDSL features (the distinction between ENTITY and PROPERTY, e.g.) have already been introduced to EXPRESS in a satisfactory way. The scope concept has been recognised as an important feature but has not been implemented yet in EXPRESS. This is the principal obstacle at present against a mapping of the CAD*I specification onto STEP.

5.1.2 Data structuring level

The need to specify semantics on the data structuring level has been presented to the ISO community working on STEP and has been generally accepted. The implementation of this approach is still underway.

5.1.3 The reference model level

The STEP development and CAD*I work are in agreement on this level.

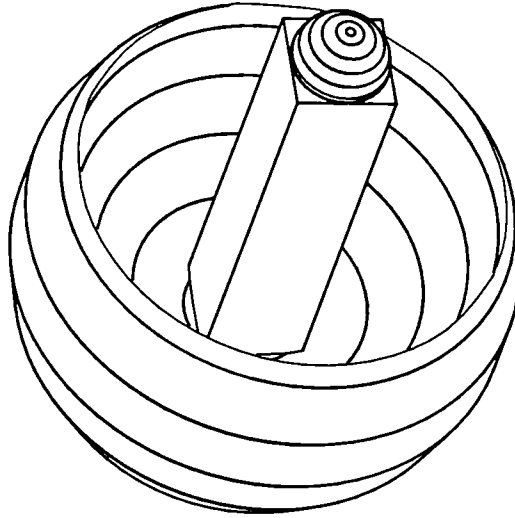


Figure 7. The "Spinning Top" test part

5.1.4 Terminology

CAD*I has developed a special terminology. CAD*I has introduced and defined terms like entity, property, attribute on the data structuring level, terms like entity-statement, property-statement for the neutral file, and terms like world, assembly, or part library on the reference model level.

The step terminology is still burdened with the abundant use of the term "entity" for all kinds of information. CAD*I is presently working on the introduction of its terminology into STEP.

5.2 GEOMETRIC MODELING

The discussion of the solids part for STEP started during a combined ISO/IGES/PDES meeting in Seattle July 1986. The CAD*I representatives presented their concept for solid model exchange to the IGES Solids Technical Committee. It turned out that this was the very point in time where the committee intended to decide on the details of a CSG part for IGES 4.0 and on a concept for a B-rep part for IGES 5.0. [Note that the IGES committee decided in the meantime to stop the development of IGES with IGES 4.0 and shift all efforts to the PDES project.] It was the feeling of the IGES Solids Technical Committee that it should avoid duplicate work and specify the solids concepts in such a way that they could be used for STEP as well. As stated above the CSG part of IGES 4.0 is derived from ESP. A reference schema has been created from that proposal and is currently under discussion in the STEP Logical Layer. It is enhanced by the capability of symbolic referencing in the definition of entities where the original IGES schema allows only for static values. Additionally the CAD*I representatives proposed to replace the rather restricted representation for the Solid of Revolution and the SOLID of Linear Extrusion by the respective concepts in the CAD*I specification, the SWEEP entities, and to use the SCOPE mechanism.

The B-rep was still under discussion in that meeting. Three concepts were proposed as potential candidates for the inclusion in IGES and STEP:

```

CAD*I_FORMAT_BEGIN_19851011_METAFILE
CAD*I_FORMAT_BEGIN_19860611_EUCLID_PREPROCESSOR
HEADER('GENGENBACH/REICHERT','Kernforschungszentrum Karlsruhe','Micro-VAX II','V
MS_4.2','EUCLID-Preprocessor 1.0','7-MAY-1987','??disclaimer??','4','3','0','0
',10,5,5);WORLD(OPEN);WORLD_HEADER(+1.00000E-03,+1.00000E+00,+2.00000E+03,+1.0000
0E-04);SCOPE;CONSTRUCT(#1:OPEN);SCOPE;SOLID_SPHERE(#5:+1.50000E+02,PLACEMENT(:RO
T_MATRIX(:DIRECTION(:+1.00000E+00,+0.00000E+00,+0.00000E+00:),DIRECTION(:+0.0000
0E+00,+1.00000E+00,+0.00000E+00:),DIRECTION(:+0.00000E+00,+0.00000E+00,+1.00000E
+00:)),POINT_CONSTANT(:+1.00000E+02,+1.00000E+02,+1.00000E+02:));SOLID_SPHERE
(#6:+1.30000E+02,PLACEMENT(:ROT_MATRIX(:DIRECTION(:+1.00000E+00,+0.00000E+00,+0.
00000E+00:),DIRECTION(:+0.00000E+00,+1.00000E+00,+0.00000E+00:),DIRECTION(:+0.00
00E+00,+0.00000E+00,+1.00000E+00:)),POINT_CONSTANT(:+1.00000E+02,+1.00000E+02,
+1.70000E+02:));BOX(#7:+6.00000E+01,+6.00000E+01,+3.00000E+02,PLACEMENT(:ROT_M
ATRIX(:DIRECTION(:+1.00000E+00,+0.00000E+00,+0.00000E+00:),DIRECTION(:+0.00000E
+00,+1.00000E+00,+0.00000E+00:),DIRECTION(:+0.00000E+00,+0.00000E+00,+1.00000E+00
:)),POINT_CONSTANT(:+7.00000E+01,+7.00000E+01,+0.00000E+00:));SOLID_SPHERE(#8
:+3.00000E+01,PLACEMENT(:ROT_MATRIX(:DIRECTION(:+1.00000E+00,+0.00000E+00,+0.000
00E+00:),DIRECTION(:+0.00000E+00,+1.00000E+00,+0.00000E+00:),DIRECTION(:+0.00000
E+00,+0.00000E+00,+1.00000E+00:)),POINT_CONSTANT(:+1.00000E+02,+1.00000E+02,+3.
00000E+02:));BOOLEAN(#4:UNION,#7,#8);BOOLEAN(#3:DIFFERENCE,#5,#6);BOOLEAN(#2:U
NION,#3,#4);END_SCOPE;CONSTR_RESULT(#2);CONSTRUCT(#1:CLOSE);END_SCOPE;WORLD(CLOS
E);
CAD*I_FORMAT__END_19860611_EUCLID_PREPROCESSOR
CAD*I_FORMAT__END_19851011_METAFILE

```

Figure 8. The CAD*I CSG file of the test part "Spinning Top"

- the ESP B-rep,
- the PESTT proposal from Peter Wilson [11],
- the CAD*I B-rep proposal.

The ESP B-rep was discarded because the experiences with XBF to which it is rather close did not speak in favor of it. The remaining two proposals PESTT and CAD*I turned out to have quite a lot in common. The CAD*I group and the PDES group used the subsequent ISO meetings in Frankfurt and West Palm Beach to negotiate about the details of a common solution. It seems feasible to come to that solution in this year.

In addition to those two concepts the CAD*I team brought its POLYHEDRON representation into the discussion. That concept was generally accepted at the West Palm Beach ISO meeting April 1987.

The working group 1 (Wireframe models) and working group 3 (Surface models) of the CAD*I project are in charge of the development of the curves and surfaces proposal for STEP.

5.3 APPLICATION LAYER

The specification of the CAD*I FEM group strongly influences the development of the FEM reference model on the Application Layer. The CAD*I specification for analysis data forms a basis for the CAD*I contribution [19].

5.4 PHYSICAL LAYER

The basic concepts of the CAD*I physical layer have already significantly influenced the STEP development. The basic alphabet, the method for encoding national characters and the token syntax, have been accepted in the specification of the physical file format of STEP. The syntax of entity statements is not identical for CAD*I and STEP, but there is a one-to-one mapping. The syntax difference is illustrated by the following example:

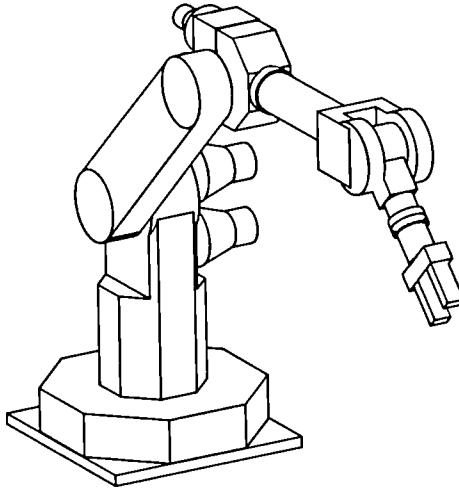


Figure 9. A more complex CSG test part "EU/robot"

```

CAD*I:
  POINT( #1 : 0.1 , 0.6 , -3.5 );
  LINE_SEGMENT( #2 : #1 , POINT( 0.8 , 0.6 , -3.5 ) );

STEP (not finally specified):
  @1 = POINT/ 0.1 , 0.6 , -3.5 ;
  @2 = LINE_SEGMENT/ #1 , POINT/ 0.8 , 0.6 , -3.5 ; ; !

```

Another concept developed by CAD*I for the physical file is the "envelope concept". It represents a method for sequencing in a single sequential file (called CAD*I metafile) any number of neutral files which were written according to different format specifications. This technique is useful for grouping in a single file information of different kind: a text description of a product, a computer graphics metafile of the same product, experimental data resulting from tests, and a CAD*I geometry model. The approach allows for "peaceful coexistence" of different standards on the same file. This concept has similarities to the "viewtype" approach developed in ESPRIT project 688 in conjunction with ISO/TC184/SC5 [20].

So far, the STEP community could not be convinced to include this concept in the STEP specification.

6. CONCLUSION

The various attempts to establish a transfer standard for solid model exchange reflect on one hand the continuous development in the solid modeling area and on the other hand the effect of the deficiencies inherited from the early concepts such as IGES.

That experience also gained in other CAD/CAM areas led to the perhaps most important resolution of the STEP effort: No existing standard will become STEP. However that does not mean to discard all the experiences gained so far. On the contrary it is the unique chance for all countries represented

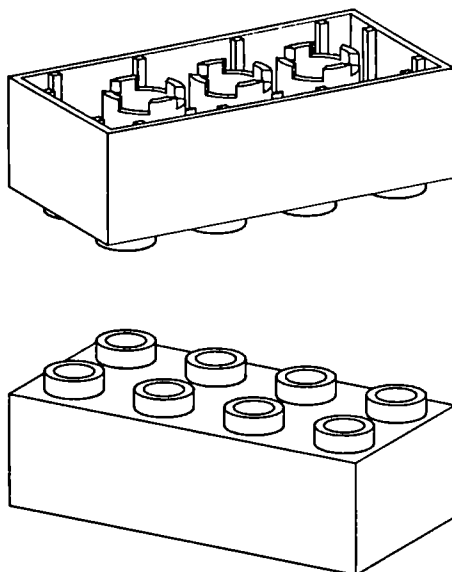


Figure 10. A basic B-rep test part "Lego brick"

in the ISO to bring into the discussion their specific knowhow without being restricted by some existing standard or concept.

On the other hand most of those concepts mentioned above were mainly developed in the US. So far the European contribution was rather small compared with the manpower allocated to those topics in the US. Thus, IGES was an almost entirely US development.

The CAD*I project has changed that situation for the development of STEP. Its intent is to unite and enlarge the experience present in the different countries and thus build up a European position in the area of CAD data exchange.

The CAD*I project will continue processor implementation and exchange of test files thus gaining experience to refine and enhance the specification. At the end of the project not only a consistent specification for CAD data structures and for their mapping on a neutral file will be available: there will also be a set of pre- and post-processors for a number of representative commercial CAD systems. Both the specifications and the processors will be closer to the still evolving STEP standard than they are today. As a strategy, the CAD*I project will adopt STEP formulations wherever the experience from processor development and CAD data exchange tests indicate that such a move is advisable. Where STEP deficiencies can be identified from CAD*I work the project will make every effort to eliminate these deficiencies via the participation of its members in the national and international bodies of standardisation.

CAD*I will, however, not limit its scope to what STEP is concerned with at present. It will emphasize topics that are not yet treated on the STEP level such as parametric models and macros and make sure that the presently limited scope of STEP does not introduce limitations in its implementation which will later make it difficult to standardize the transfer of more sophisticated product models (such as standard parts) according to needs of European industry.

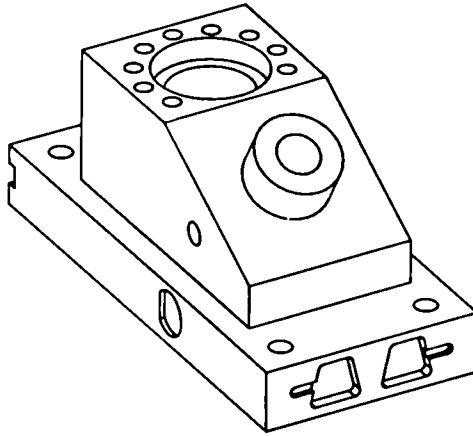


Figure 11. The B-rep test part "ANC 101"

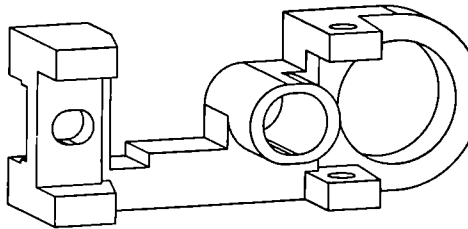


Figure 12. The B-rep test part "Gehaeuse"

7. REFERENCES

- [1] I. Bey, J. Leuridan: ESPRIT Project 322, CAD Interfaces: Interfaces in CAD Systems. presented at the ESPRIT Technical Week 1985.
- [2] I. Bey, J. Leuridan (editors): Development of Specifications for Exchange of Product Definition and Analysis Data. ESPRIT Project 322, CAD*I: CAD Interfaces. presented at the ESPRIT Technical Week 1986.
- [3] I. Bey, J. Leuridan (Editors): ESPRIT Project 322: CAD*I (CAD Interfaces) Status Report 2. (1986), Kernforschungszentrum Karlsruhe.

- [4] I. Bey, J. Leuridan (Editors): ESPRIT Project 322: CAD*I (CAD Interfaces) Status Report 3. (1987), Kernforschungszentrum Karlsruhe.
- [5] A.A.G. Requicha et al., Solid Modeling: A historical summary and contemporary assessment, IEEE CG&A, 9-24 (1982).
- [6] ANSI, Digital Representation for Communication of Product Definition Data. ANSI Y14.26M (1981).
- [7] DIN, Format zum Austausch geometrischer Informationen. DIN 66301 (1986).
- [8] AFNOR, Representation externe des donnees de definition de produits, Specification du standard d'echange et de transfert (SET) Version 85-08, Norme Experimentale Z 68-300 (1985).
- [9] A.A.G. Requicha, Representations for Rigid Solids: Theory, Methods, and Systems. ACM Comp. Surveys Vol. 12, 437-464 (1980).
- [10] CAM-I, Geometric Modeling Project Boundary File Design XBF-2 (1982).
- [11] P.R. Wilson et al., Interfaces for Data Transfer Between Solid Modeling Systems. IEEE CG&A, 41-51 (1985).
- [12] IGES, E.S.P. Experimental Solids Proposal (1984)
- [13] NBS, IGES Mail Ballot RFC's and CO's for Version 4.0 Document, (1986).
- [14] P.R. Wilson, Data Transfer and Solid Modeling. International workshop on standardisation in Computer Graphics, Genua (1986).
- [15] E.G. Schlechtendahl (Editor), Specification of a CAD*I Neutral File for Solids Version 2.1, ESPRIT Project 322, Springer Verlag (1986).
- [16] CAD*I project: Specification of a CAD*I Neutral File for CAD Geometry, Version 3.2 ESPRIT Project 322, Springer Verlag (1987).
- [17] D. Schenck: The data specification language DSL, (1985), unpublished.
- [18] N. Brändli, Anwendung von Compiler-Generatoren zur Generierung von Post-/Preprozessoren im ESPRIT-Projekt CAD*I, VDI Berichte Nr. 610.5, 1986, 33-41 (1986).
- [19] J. v. Maanen, D. Thomas (Editors): Specification for the Exchange of Product Analysis Data, ESPRIT Project 322: CAD*I (CAD Interfaces) Kernforschungszentrum Karlsruhe (1987)
- [20] G. N. Marechal: Towards a Reference Model for a Computer Integrated Open System Architecture. ISO/TC184/SC5/WG1 document N95 version 2, March 20, 1987 (presented at the ISO/TC184/SC4/WG1 meeting at West Palm Beach, March 30 - April 3, 1987)

Development of Advanced Modelling Techniques

H. Grabowski, R. Anderl, B. Pätzold, S. Rude

Institute für Rechneranwendung in Planung und Konstruktion,
University of Karlsruhe, 7500 Karlsruhe, West Germany

The Esprit project 322 "CAD*I" (CAD-Interfaces) is concerned with interfaces to CAD systems. One of these interfaces is the user interface, whose improvement is the goal of working group 5 (WG 5) "Advanced Modelling".

The main question involved within the WG 5 is: "*What is the language of the designer ?*", i.e. what kind of communication is required to express the ideas of a designer.

Two main approaches are pursued within the Advanced Modelling group:

One of the approaches is a language oriented approach, concerned with the question, how technical terms of a designer can be used to communicate with a CAD system. It is thought to explore the application of terms, which are used to specify some "form features", as well as terms, which are used to specify constraints between design elements, called "technical associations".

The second is a graphical oriented approach, and should clarify the question, how a designer can be supported by some application of handsketching techniques.

First essential of the realization of our approaches is an argued set of CAD functions, called basic methods, as well as an augmented model, called product model.

1. Introduction

1.1 General Background of the Project

The ESPRIT project 322 "CAD Interfaces" has been established in 1984 to define the most important interfaces in CAD/CAM systems for data exchange, data base access, finite element analysis, experimental analysis and advanced modelling techniques (fig. 1-1). The project work has entered now its third year of research and development work. The total scheduled project duration is 5 years, from November 1984 to October 1989. 120 man years are planned to reach the final goal and results, that means: a set

of powerful vendor independent interfaces with its corresponding pre- and postprocessors.

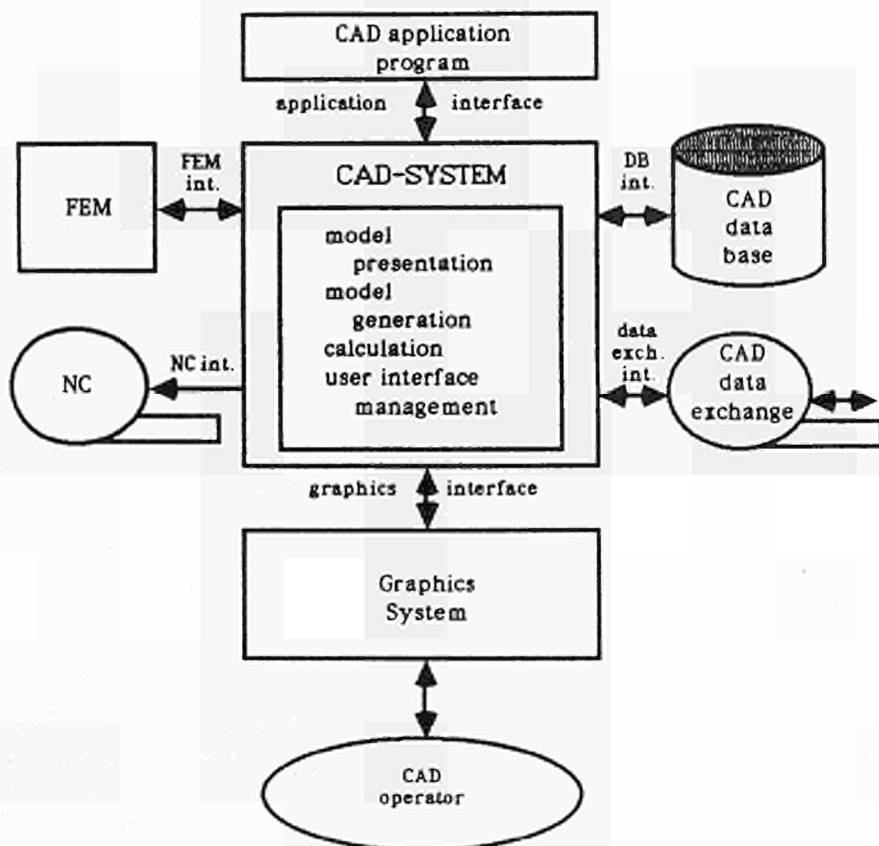


Fig. 1-1: Interfaces in CAD/CAM systems

1.2 CAD*I Project Organization

The research and development work within the CAD*I project is organized in eight working groups according to the main technical issues. The task dealing with the communication interface between a designer and a CAD system has been overtaken by working group 5.

1.3 Advanced Modelling

WG5 has been entitled "Advanced Modelling", because there were to develop future concepts to communicate with a CAD system in terms of a designer.

Therefor the overlapping question arised: *What is the language of a designer ?* Two major approaches are persued within the project as shown in fig. 1-2.

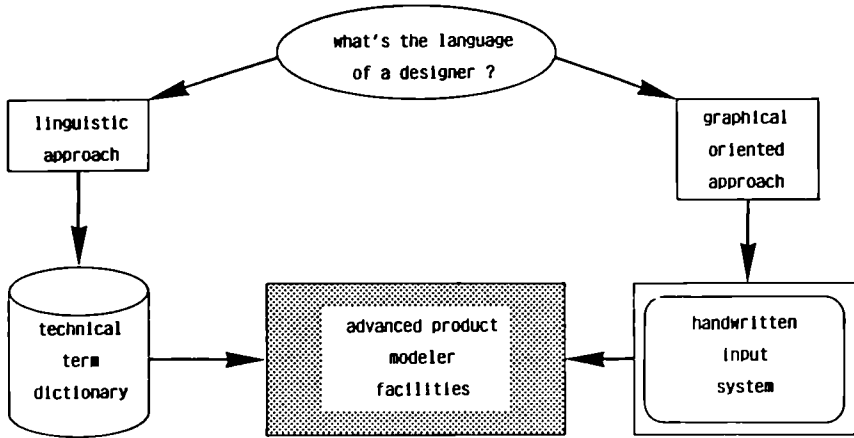


Fig. 1-2: Approaches on Advanced Modelling

One of the approaches is a linguistic one, i.e. all languages to communicate with a CAD system can be seen hereby. The other possibility is a graphical oriented approach, i.e. all communication techniques using graphical elements are meant. We would like to say that no other approaches to communicate with a CAD system exist and all efforts to increase communication performance must improve one of our approaches in principle.

To reach a higher level of communication on the graphical oriented side, we think, efforts to allow handwritten input into a CAD system must be an improvement in a sense, that not only input of sequences of points ale allowed, but also techniques to recognize graphical elements as well as constraints between themselves. On the other side input of natural language would be the best suitable goal, but a requirement therefor is to think about the semantics of technical terms.

2. Advanced Modelling Concept

A conceptual approach for an improvement of modelling techniques is based on the functional structure on CAD systems (see fig. 2-1).

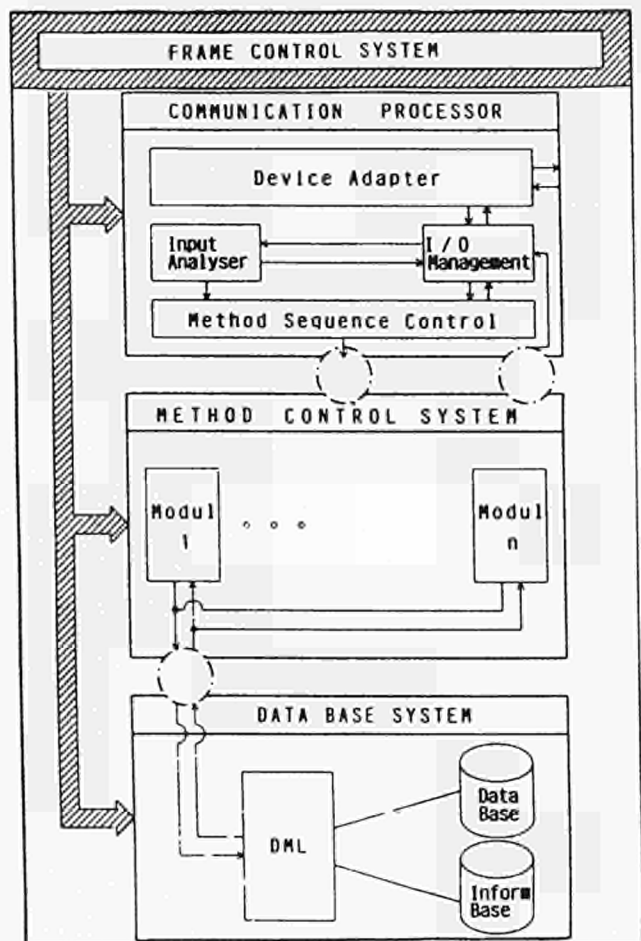


Fig. 2-1: Functional structure on CAD system

Fundamental for all applications planned with such a system is the definition of a conceptual schema, which is able to store all product definition data. Two different approaches are known: Design process oriented phase models /1,2/, or a coherent product model. We have chosen the coherent product model.

2.1 Coherent Product Model

A coherent product model covers the total amount of technical relevant product information as well as its semantical relations. To get an overview about this model, it was taken apart in logical clusters, i.e. partial models. These partial models are bounded in a way, that the entity sets of two partial models are disjunctive but the association sets are conjunctive (see fig. 2-2).

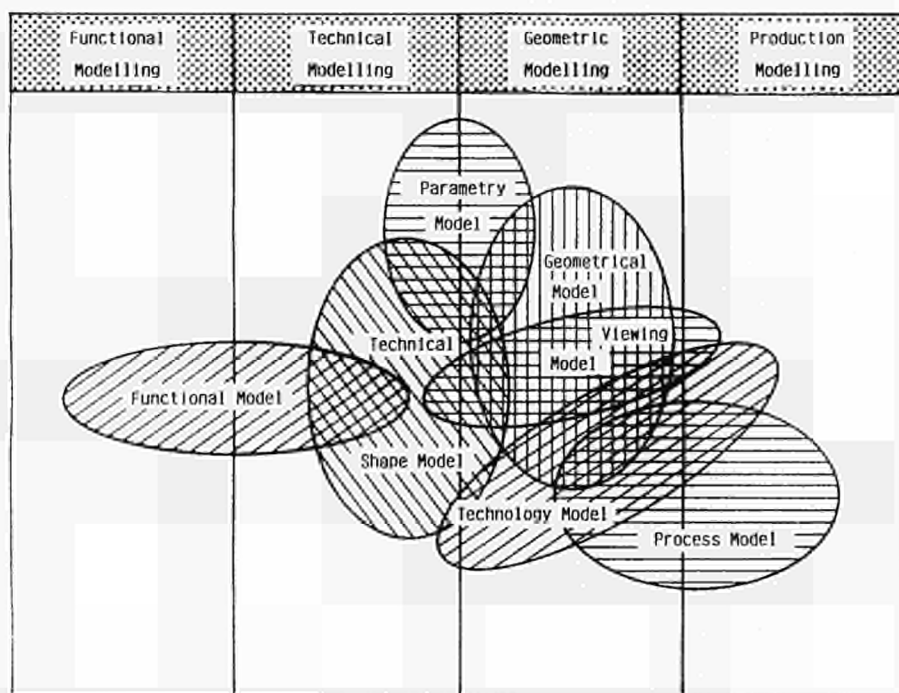


Fig. 2.-2: Venn diagram of the different partial models

Hereby an assignment to the different modelling levels, which are described in the following was done. The different partial models are:

The **Geometric Model** describes the product geometry and topology according to a boundary representation model, called macro geometry, as well as all necessary geometric data to describe additional geometry.

The **Viewing Model** documents all the information to generate views and drawings (inclusive selectionals by referencing an appropriate face set), as well as all information to process viewing specific input data.

The **Technical Shape Model** represents the shape and the structure information, which can not be described by pure macro geometry, i.e.

assembly structures, from feature entities as well as constructive spaces.

The **Technology Model** comprises the products technological data like material, tolerances, surface textures, etc. which is called micro geometry.

The **Parametric Model** contains information about the mathematical connection of product dimensions and geometric associations.

The **Functional Model** comprises all entities for modelling function structures as well as physical working principles.

Finally the **Process Model** is necessary to get the production process data by making available production related data like available machines and tools, states of parts during manufacturing process, etc.

2.2 Modelling Level Concept

Apart of this product model a modelling level concept was worked out, able to transform user input down to manipulations of the underlying data structure (see fig. 2-3).

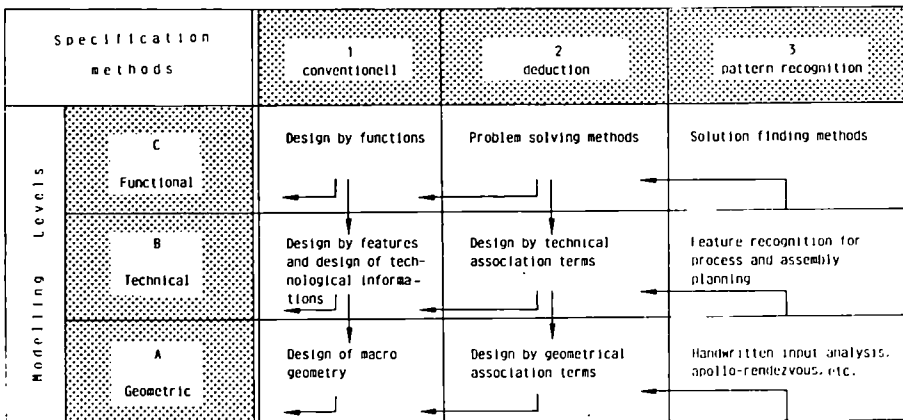


Fig. 2-3: Modelling level concept

The **Geometric Modelling Level** comprises the usage of basic methods to build up geometric models of a part. It can be distinguished the direct specification of operations together with all required parameters, the direct specification of geometric associations, and the automatically recognition of such associations.

The **Technical Modelling Level** is understood as procedures, which make use of technical input of a designer, transform it to the underlying

geometric level, but store additional information as well into other partial models. The direct specification of operations means technical elements, e.g. form features. The direct specification of associations allows the input of technical associations, like flush, etc. The indirect specification is given by an automatic recognition of technical information (e.g. form features recognition for manufacturing purposes).

The **Functional Modelling Level** must be seen in a similar way to support research, which is done in the fields of design theory.

3. Technical Terms and Handwritten Input

Based on the just presented concept for advanced modelling techniques, several applications have been worked out and partly implemented to show the feasibility of our concept.

3.1 Design by Technical Terms

This is an example for a modelling method of the technical modelling level. Semantics of technical terms can be expressed by some context conditions and actions which must be executed when the conditions are true. Technical terms defined in that way need a collection of rules to capture the semantic. Applied to our concept there must be also found a mechanism to transform the rules to internal interfaces of CAD systems. Such a mechanism could be found by taking parts of the rules as our so called basic methods and by transforming in that way complex term definitions down to a set of basic methods. Basic methods are defined in a sense, that "basic methods are the only valid functions which have access to the underlying model".

Figure 3-1 gives an example showing the working principle of our approach using the technical term "flush". On the left side there is given an application example for the use of the term within an input command by a CAD system. Result of the command should be a bolt, which is positioned and oriented within the fork in the right way. Its length should be determined by the fork as shown in the picture. In the middle of the figure is verbally described, what is the semantic of the technical term: "Flush" is an association term between two faces (<dig1> and <dig2> must identify themselves) of different parts. If the type of the surfaces is "plane" and the plane data available some actions can be executed on the model.

plane data are:

Aufpunktvektor (AVE) = arbitrary point of the plane
 Normalenvektor (NVE) = direction vector of the plane

The consequent actions on the model are follows:

Insertion of the technical association flush between the two surface entities, making the direction vectors (NVEs) identical, i.e. giving the planes the right orientation, and taking care to make the distance between the planes zero, i.e. the result must be, that the two points of the surfaces lie both within the same plane and the difference vector is therefor perpendicular to the normal vector of the planes (Remark: If the distance vector is zero and the direction vectors are made identical; then the planes are also identical).

The right part of the figure shows the formalization of the verbal specification which is necessary to compute the defined semantic: Verbal specifications are transformed to rules, which have a condition and an action part. Every single condition as well as an action refers to a basic method, which is called "procedure" and which has access to the underlying product model. Having access to the product model, modifications are possible which lead to future restrictions for subsequent modelling steps.

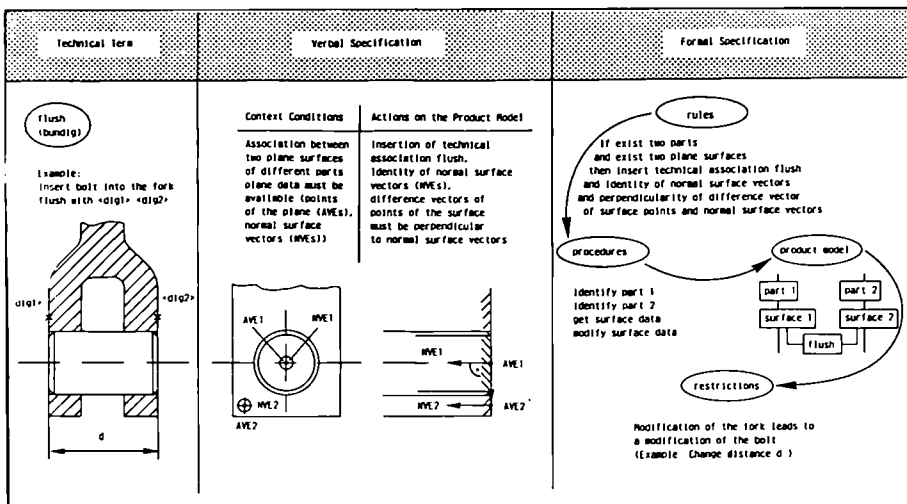


Fig. 3-1: Transformation concept applied to technical terms

Fig. 3-2 gives some more examples which show the result of commands containing each a technical association term, and therefor a lot of technical semantics which can be used to determine the shape of objects to be described by a designer.

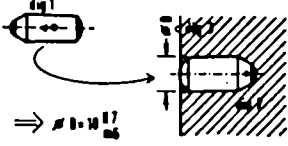
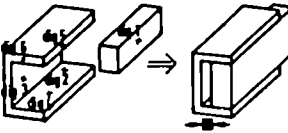
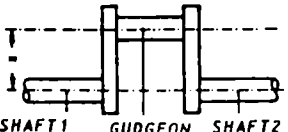
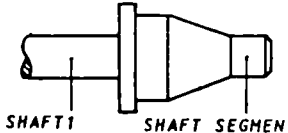
TECHNICAL DESIGN TERM	COMMAND EXAMPLE	APPLICATION EXAMPLE
<i>fitted in</i>	INSERT plain pin DIN 6325 <dig1> INTO hole <dig2> FITTED IN D=H7, m6 FLUSH <dig3>	
<i>flush</i>	INSERT bar <dig1> INTO clip <dig2>, DISTANCE 10 FROM <dig3>, FITTED IN <dig4>, FLUSH WITH <dig5> <dig6>	
<i>axially parallel</i>	INSERT gudgeon AXIALLY PARALLEL WITH DISTANCE x TO shaft1 AND shaft2	
<i>coaxial</i>	INSERT shaft segment COAXIAL TO shaft1	

Fig. 3-2: Application of technical association terms

3.2 Handwritten Input System

As an example for modelling techniques using pattern recognition can serve our method to recognize handsketched input. The handwritten input system is able to analyze and to interpret the input based on two main components: Pattern recognition, and context analysis, which is shown in fig. 3-3.

It is a user oriented communication technique which is close to the conventional working method of designing.

The main Tasks of a Handwritten Input System are as follows:

1. The user defined amount of information like sketched geometric entities respectively geometric symbols have to be recognized in their syntactical meaning.

2. Since the user input is in a semantical connection to previous input or to the state of model it has to be interpreted with respect of its semantical context.
3. The recognized syntactical entities (geometry, symbols) and its semantical context have to be stored in a computer internal model to be available for further semantical analysis.

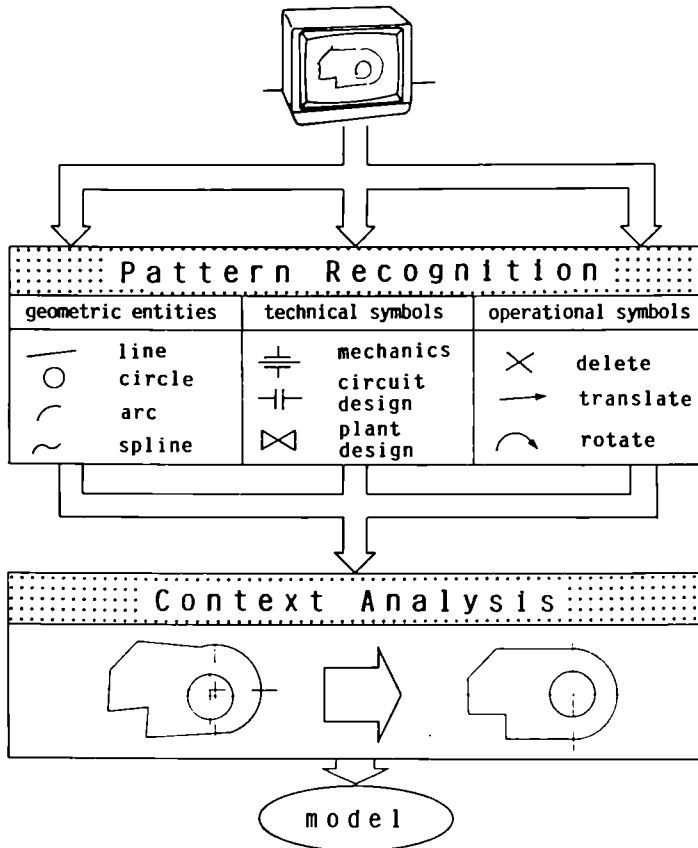


Fig. 3-3: Structure of the handwritten input system

An object can be described by a set of geometric entities existing of type and this position, orientation and dimension (POD) parameters and by a set of associations between these entities. Geometric associations are constraints between geometric entities and their POD-parameters.

Geometric associations can be defined by the user or by the geometric context. In the Context Analysis System the determination of the geometric association is based on the geometric context (see fig 3-4).

Following geometric associations are distinguished

- parallelism
- angularity
- co-axiality
- symmetry

These geometric associations have with respect to the geometric type of entities (line, circle, arc) different semantic. The semantic is described for some entities in form of a rule base. Fig. 3-4 shows an excerpt of the rule base.

Every rule exists of a test- and an action-part. The testpart describes the conditions and the actionpart a set of operations which must be performed regarding the model.

GA	rule	example
//	<pre> par(i,j) := [i:typ(LINE), j:typ(LINE), win(i,j) <ξ, => ins_par(i,j)], [...]. </pre>	
∠	<pre> win(i,j) := [i:typ(LINE), j:typ(ARC), tangente(j,P,k), win(i,k) <ξ, => ins_tan(i,j)], [...]. </pre>	
⊙	<pre> koax(i,j) := [i:typ(CIRCLE), j:typ(CIRCLE), abs(M(i),M(j)) <ξ, => ins_koax(i,j)], [...]. </pre>	
≡	<pre> sym(i,j,k) := [i:typ(ARC), j:typ(ARC), k:typ(SYM_LINE), abs(i,k)-abs(j,k) <ξ, => ins_sym(i,j,k)], [...]. </pre>	
<p>legend:</p> <p>i - existing entity j - new entity k - tangent or axis of symmetry P,M - points</p>		

Fig: 3-4: Rule base of the semantic context analysis system

4. Conclusions

The goal of the working group 5 within the CAD*I project is the development of a concept for advanced modelling. Requirements for such a concept were derived from the principal possibilities to communicate with a CAD system by language or by graphical input.

Within the functional structure of a CAD system, first essential of the concept is a coherent product model. Corresponding to it, there is required a well structured method system which has been found by the definition of a modelling level concept.

We have started to implement two well-chosen examples to verify the advanced modelling concept. Result of this work is the finding, that for the development of advanced modelling methods the internal structure of a method system must be built up in an hierarchical way. The methods can be classified in different modelling levels. To realize the concept the definition of internal procedural interfaces will be the main task of the future (see fig. 4-1).

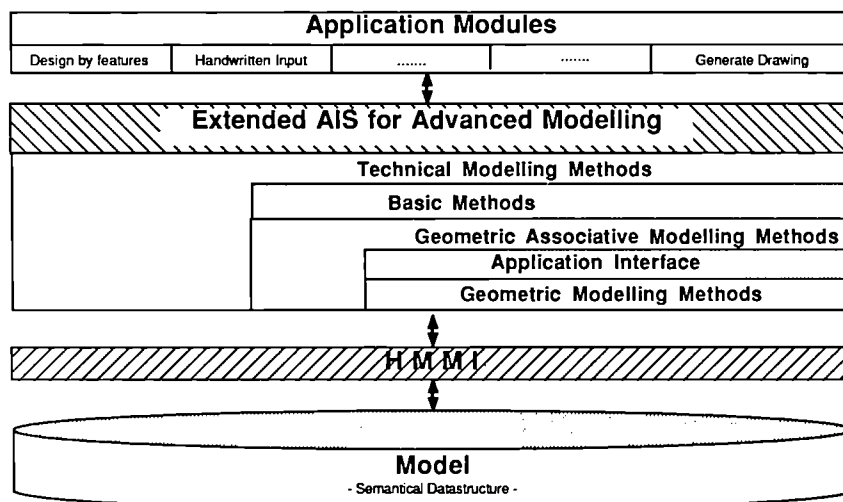


Fig. 4-1: Specification of a internal interfaces

5. References

- /1/ Krause, F.-L., Methoden zur Gestaltung von CAD-Systemen, Dissertation, Berlin, 1976
- /2/ Tomiyama, T., ten Hagen, P.: Organization of design knowledge in an intelligent CAD environment, in Expert Systems for Computer-Aided Design, ed. J. Gero, North - Holland, Amsterdam, 1987.

- /3/ Grabowski, H., Anderl, R., Rude, S.: An Overview for a Concept on Advanced Modelling, WG5 Report, ESPRIT CAD Interfaces (CAD*I) Project, Kernforschungszentrum Karlsruhe, 1986.
- /4/ Grabowski, H., Anderl, R., Rude, S.: Technical Term Dictionary with Related Geometric and Technical Semantics, WG5 Report, ESPRIT CAD-Interfaces (CAD*I), Kernforschungszentrum Karlsruhe, 1986.

Project No. 496

PAPILLON - A SUPPORT ENVIRONMENT FOR GRAPHICAL SOFTWARE
DEVELOPMENT*

Chris Chedghey

Generics (Software) Ltd., 7 Leopardstown Office Park, Foxrock, Dublin
18, Ireland.

The Papillon project (ESPRIT 496), entitled "A configurable graphics subsystem for CIM", is now in its final year. A prototype system has been produced which helps the application developer build well-structured software which is initially devoid of representation or user-interface information. Graphical representations for application data types may be subsequently described without direct coding. Calls made to the application by "driver" software such as process control or simulation programs will now be reflected in the display. A user interface may be automatically generated to enable a human user to "drive" such applications as production planning and scheduling. The approach has its foundations in Object-Oriented Design (OOD), the Vienna Development Method (VDM) and the programming language ADA*.

1. INTRODUCTION

The aims of the Papillon project are to produce a graphical subsystem for Computer Integrated Manufacturing (CIM) which is configurable with respect to differing hardware, application and user interface requirements.

Configurability for different hardware has been largely addressed by implementing a skeletal Graphical Kernel System (GKS) which may be instantiated with comparative ease for different hardware (this work has been undertaken by GTS/GRAL GmbH, Darmstadt). The rest of the subsystem is based on GKS and may consequently be ported to different hardware environments with minimal effort.

This paper deals exclusively with the part of the subsystem which lies above the GKS and addresses the configurability for different applications and user interfaces. The subsystem is primarily intended to benefit the CIM application developer by easing the task of generating new applications, removing the need to write user interface software, and providing the facility to tailor the user interface to individual user requirements.

* The work reported here is partly funded by the Commission of the European Communities under the Esprit Programme.

* Ada is a trademark of the U.S. Government, AJPO.

While the underlying principles of the subsystem are largely application independent, the facilities supported by the prototype are aimed at one main section of the CIM application domain.

2. OBJECT-ORIENTED DESIGN, VDM META-IV AND ADA

These three orthogonal technologies have been combined into a powerful design and development method. This method was used to develop the prototype and its influence will become apparent from later sections of the paper.

2.1. Object-Oriented Design and Ada

Object-Oriented Design (OOD) is a system design methodology which is becoming increasingly popular in both software [Booch 83] and hardware [Organick 82] design.

Traditional control-based software has been data/procedure oriented, whereby programs are composed of a set of data and a set of procedures. With this approach, much of the expected benefit of decomposition is not realised because of the mutual dependency of many subprograms on the form and integrity of common data [Robson 81].

With the object-based approach to software a single type of entity, the object, represents both the data and procedure entities. Data structures are associated with a fixed set of subprograms which are the only operations defined on the object. Subprograms bodies, dependencies and implementation details are hidden. Such entities are sometimes called Abstract Data Types (ADT's) or Abstract Objects.

Programming languages which provide encapsulation and abstraction facilities are particularly suitable for this type of object-oriented programming. Ada provides these facilities through the "package" construct.

2.2. VDM Meta-IV and Ada

The Vienna Development Method (VDM) is a well established formal development method [Bjorner 78] [Jones 83]. At the kernel of the method is a formal specification language, one dialect of which is called Meta-IV [Bjorner 80]. Based on sound mathematical foundations, Meta-IV has proved itself in a number of important software developments such as the DDC Ada Compiler.

Meta-IV is a model-oriented language, providing a carefully selected collection of data types out of which new types may be modelled. The semantics of the operations on the new data types is then specified in terms of operations on the basic Meta-IV types. It is possible to implement these basic Meta-IV types as ADT's in Ada [Mac an Airchinnigh 87].

3. THE EXAMPLE APPLICATION

The chosen application domain is that of Production Planning and Scheduling (PPS) and a very brief description of this is given here.

The user of the application is the manager of a production unit in a discrete manufacturing plant. The manager receives orders from customers which he must fulfill using a pool of resources available to him. He wishes to allocate the resources in such a way that they are as near to being fully utilised as possible, but not being over utilised.

The manager will wish to input a plan for each order in the form of a Pert chart, with the nodes of the chart representing individual activities and edges indicating which activities must be completed before others can commence. He will also wish to view the activities in the form of a Gantt chart with the current time line displayed. In order to enable him to achieve optimum resource allocation he will require to examine the resource loadings as bar charts. Depending on the managers task in hand, he may require different pieces of information displayed on the screen, and he should have maximum control over this.

In the following sections, the mechanism by which this application may be constructed is outlined. Figure 1 shows the various components of the Papillon subsystem and the structure of the generated run-time system and the reader should occasionally refer to this diagram.

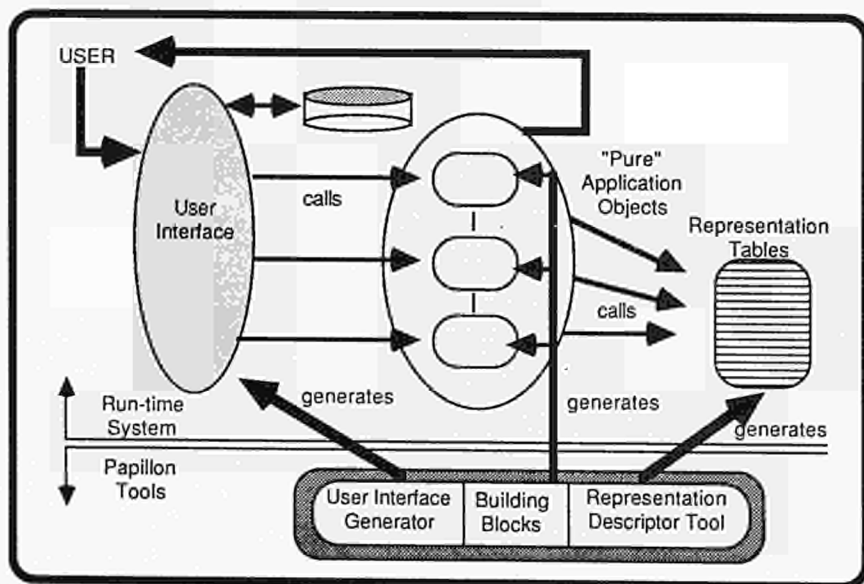


FIGURE 1

3.1. The Papillon Building Blocks

Experience has shown the Meta-IV types to be powerful building blocks in system design. The Papillon Building Blocks are a collection of packages, generic packages and package generators which enable Abstract Data Types to be constructed which are based on, though somewhat more elaborate than, these Meta-IV types.

The basic types supplied are: integer, character string, real and token. The composite types, which may be constructed from one or more subtypes, are set, list, record, map, and type union. All of these types, with the exception of perhaps the last two, will be familiar to most programmers. A map is a finite mapping from a domain type to a range type and may be thought of as a set of domain-range pairs. A type union is a type which may take values of a number of different types.

3.1.1. The construction of the application software

Figure 2 shows how the "pure" application object types may be easily modelled using the Papillon Building Blocks. The object types are presented in hierarchical (top-down) order. Although Meta-IV is used as a convenient notation in figure 2 it is unnecessary to have any knowledge of VDM in order to use the Papillon tool. The following brief description should adequately explain the implementation.

```

Order_plans = STRING m-> Order_plan
Order_plan = Activity m-> Connection_set
Connection_set = Connection-set
Connection :: s_delay : INTEGER
             s_activity_id : STRING
Activity   :: s_id      : STRING
             s_dir     : INTEGER
             s_desc    : STRING
             s_start   : Date
             s_end     : Date
             s_resources_used : Resource_descriptors
Resource_descriptors = Resource_descriptor-set
Resource_descriptor :: s_resource_id : STRING
                    s_amount : REAL
Resource_pool = Resource-set
Resource :: s_id : STRING
          s_amount_available : REAL
          s_requests : Request_set
Request_set = Request-set
Request :: s_start : Date
         s_end : Date
         s_amount_requested : REAL
Date :: s_day : INTEGER
       s_month : INTEGER
       s_year : INTEGER

```

FIGURE 2

The type `Order_plans` is a mapping from unique identifiers onto order plans. An `Order_plan` is a mapping from activities to the set of dependent activities (activities which cannot commence until this activity has been completed). A `Connection_set` is a set of connections. A `Connection` is a record containing the time before the dependent activity can commence and the identifier of the dependent activity (these fields may be selected by the functions "`s_delay`" and "`s_activity_id`"). An `Activity` is a record containing a unique identifier, a duration, a textual description, a start and end date and a set of resources required by the activity.

The meaning of the remaining object types should now be clear to the reader. Once the structure of the application object types has been decided they can be implemented directly using the Papillon Building Blocks, without the need for "coding" as such.

Once the basic types have been implemented, some will require an additional, higher_level package or "semantic layer" in order to perform checks on parameters and/or to replace or extend the interface to the object type. For example not all triplets of integers create a valid date, and an interface to the `Order_plan` type containing such operations as add activity, delete activity, schedule, etc is required rather than the interface to the basic map type with operations such as merge, domain, apply etc. The higher level operations are implemented as calls to the operations provided by the basic types.

These extra packages then contain the "pure" application algorithms, the semantics relating directly to the essence of the application in hand, and no more. In fact these are the only packages which require actual programming.

When the application packages have been created they do not do anything themselves, rather they are available for use by any "driver" software which may wish to create and manipulate instances of the data types.

3.2. The Representation Descriptor Tool (RDT)

This is a software tool which allows the user interface designer, who may also be the application developer, to describe a number of graphical representations for each application object type and to enter them into the representation tables (see Figure 1). Possible representations vary according to which of the Building Block types the application type is constructed from, and the representations which have already been defined for its sub-object types.

A window is opened on the display surface in which the user interface designer may describe representations largely through mouse input. A "fixed" piece of graphical data may be associated with any object type representation. This image will be displayed for all objects of that type. If the object type has sub-objects then positioning rules, size and representation identifier (referring to a representation already in the tables) may be given so that the particular sub-object instances may be displayed as part of the parent object display.

Because of the hierarchical nature of object type representations, it is usual to start with the objects lowest in the hierarchy and work upwards. In our

example we would start with types INTEGER, STRING and DATE. For such low-level and frequently used types, there would normally be a number of previously defined (standard) representations in the tables, though it is always possible to define new ones. The type of information which may be given to describe representations for these types includes size, font, direction, surrounding boxes etc.

Two possible representation templates and corresponding instances for the Activity type are shown in figure 3. When we come to describe representations to generate an easy-to-read Pert chart and the other for a detailed chart. It is possible to specify that the user locate the Activities in the Order_plan (Pert chart format) or that they are listed in rows and columns, perhaps in a specified order (directory format). We can also specify a bar chart representation for the Order_plan type. One of the possible representations for the Resource type is a histogram showing the resource loading over time.

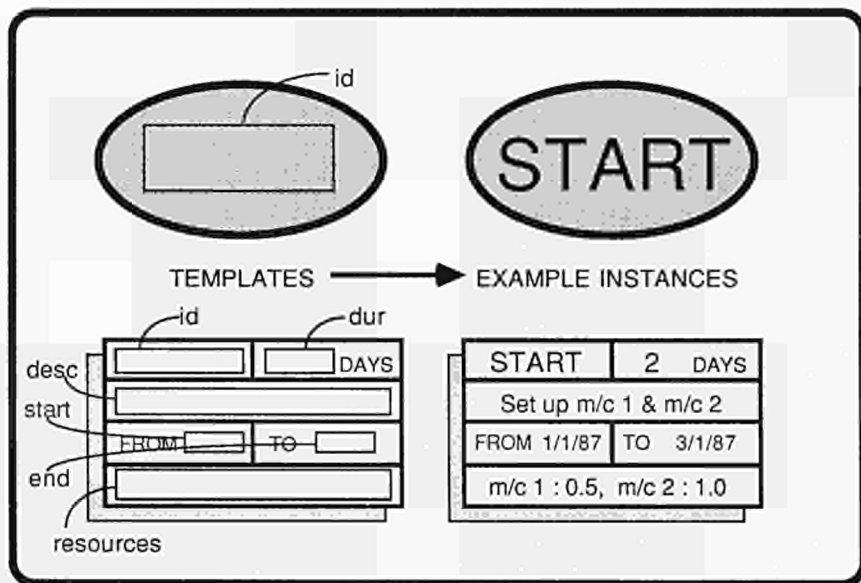


FIGURE 3

The example representations mentioned in the previous paragraph are intended only to give the reader a feel for the degree of freedom in the design of representations, and indicate just a subset of the possibilities.

3.3. The User Interface Generator

Once representations have been defined for any application object type, it is possible for instances of the type which are created by "driver" software to be displayed (though the driver software need not be aware of this). As the

driver software modifies objects, their representations will be modified accordingly.

Many applications, including that of the example, require that a human user be able to "drive" the application. The User Interface Generator creates a handler which allows a user to create, manipulate and observe instances of the application object types. The result is a mouse/window/menu - driven interface which will be described in more detail in the next section.

3.4. The Run-time System

A possible snap-shot of a run-time system generated for the example application is depicted in a simplified form in Figure 4. The arrows in the figure indicate relationships between the windows and do not appear in the actual display.

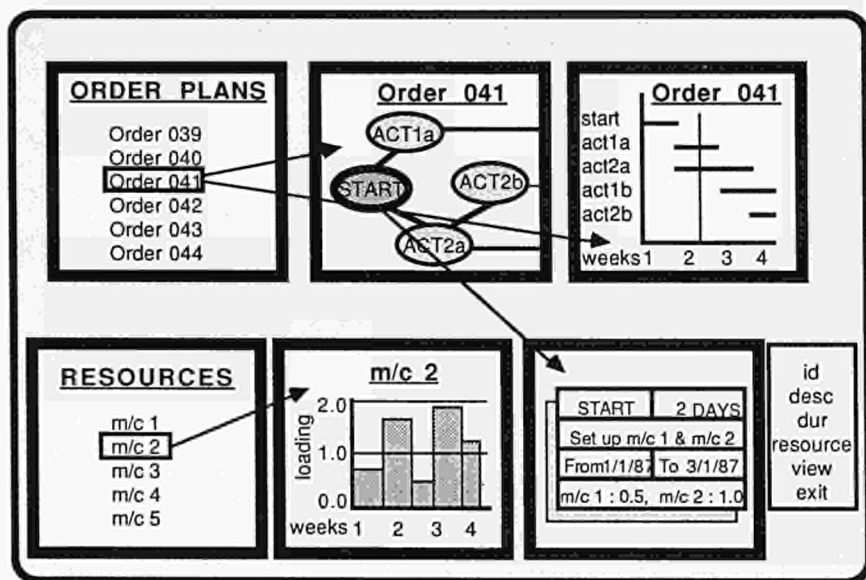


FIGURE 4

The user has loaded up the order plans and associated resources. Both of these have been chosen to be displayed in a dictionary format. He has then chosen to view one of the order plans (Order 041) in both Pert chart and bar chart format. When he modifies the order plan, both representations will reflect the change. He has then chosen to observe/edit one of the Activities in the plan ("START") and this has appeared in detail in its own window. At present this is the window into which commands are directed and the menu of available commands for Activities is displayed.

The user has also chosen to view one of the resources (m/c 2) in the form of a histogram. As order plans are modified and the resource loadings for m/c 2 change, the histogram will also change.

The user may switch between windows; for example, from working with activity "START" he may move to resource "m/c 2". He may change the representation of an object or open a new window with another representation. He has other viewing operations available such as panning and zooming on the Pert chart or shifting the displayed time slot on the Gantt chart or histogram.

When the user has finished observing/editing the activity, the activity window will close and the order plan will be updated. When he has finished with the order, the order windows will close. Eventually, when all windows have been closed, the user will be able to transfer the modified Order_plans and Resource_pool to permanent storage and perhaps load some other objects for modification or observation.

4. COMMENTS ON THE PAPILLON PROTOTYPE

4.1. Reconfigurability of the user interface

This is an important aspect of the prototype. Because the user interface software is independent of the application software, the representations of object types may be edited, replaced or added to at any time (concievably even at run-time). It may occasionally be advantageous to supply the Papillon tools with the application code in order that the user interface may be modified by OEM's or even end users to suit their needs as closely as possible.

4.2. Suitability for different applications

An analysis of the type of applications for which the Papillon tools are most suitable will require more experimentation. However, it can be expected that the approach will work best for those applications which are amenable to Object-Oriented Design. Such applications tend to be data-oriented rather than control-oriented.

Although the prototype currently supports the various object representations associated with a particular application domain, many of these representations will also be relevant to other applications. However, in order that a derived tool be useful for a wide variety of applications, a set of representations must be provided which fully supports each application.

A certain amount of investigation has been undertaken which indicates the particular suitability of the tools to model-oriented applications such as process control or simulation or a combination of both. In such applications, data-structures (objects) could be easily constructed (and given representations) to model real world entities and be manipulated by the process control or simulation software. These models and their corresponding representations could then be easily modified or augmented as the real-world situation changed, for example with the addition of new machines or machine types.

4.3. Comparisons with User Interface Management Systems (UIMS)

The work undertaken on the Papillon project is sometimes compared to that on User Interface Management Systems. There are however some important differences between the Papillon toolset and the more traditional type of UIMS [Pfaff 83].

In a UIMS, the user interface designer usually describes the user-machine dialogue and the overall form of the user interface in considerable detail by means of a fairly complicated description language. The application implementor then either "fills in the blanks" where the UIMS will call application routines, or writes his applications software to call the UIMS when input or output is required.

This approach means that the application implementor must write a certain amount of user-oriented software, although this will be minimised by the UIMS. There is generally little encouragement by the UIMS towards the generation of well-structured software which is likely to enhance reliability and reuse. Also, one can generally say that the tighter the coupling between UIMS and the application software, the less the ease of configurability of both the application and the user interface.

With the Papillon approach on the other hand, the application implementor makes no user-oriented calls, rather he is free to concentrate on the pure application algorithms. He is encouraged and indeed helped to generate his software quickly and in the form of well-structured, reusable and easily reconfigured components. As there is complete separation of application from user interface, the configurability of the user interface is maximised.

5. ENHANCEMENT OF THE TOOLSET

During the remainder of the project and during the subsequent exploitation phase, a number of advancements will be made to the toolset. Some of the envisaged areas of improvement are now mentioned.

The number of possible representations for the various Building Block types will be expanded. The available Building Block types themselves may be expanded upon to provide new types or higher level composites of existing types.

The Representation Descriptor Tools could be improved to provide more checks for valid representations and provide more assistance to the user interface designer. The facility for specifying and editing prompt and error messages should be provided. Another useful feature would be the support of generic representations for generic object types.

As is typically the case, there are a number of important areas for improvement of the generated user interface. Examples include the incorporation of more direct manipulation, the improvement of error handling, the introduction of "explosion levels", and the ability of the user to create his own commands as a sequence of existing commands.

The relative importance of the various possibilities for improvement are becoming clearer with experimentation.

6. CONCLUSIONS

One important development (and certainly the most visible) which has arisen from the Papillon project is a prototype configurable, graphical production planning and scheduling system. However, in many ways the most exciting outcome is the technique which renders the prototype application inherently configurable and expandable and which is expected to lead to many further developments as the project moves into its exploitation phase.

ACKNOWLEDGEMENTS

I would like to thank Paul Hickey, also of Generics, for his help during both the design and implementation phases of the project, the various members of Trinity College Dublin for their work on the implementation of the Representation Descriptor Tool, and Signal Computer GmbH and GTS/GRAL GmbH for their work on sections of the project not directly mentioned here. I would also like to express my sincere gratitude to Rainer Zimmermann, project officer for the CEC, for his continued help and advice throughout the project.

REFERENCES

- [Booch 83] Booch, G., *Software Engineering with Ada*, Benjamin/Cummings, 1983.
- [Bjorner 78] Bjorner, D., and Jones, C.B., (Eds.), *The Vienna Development Method : The Meta-Language*, Lecture Notes in Computer Science, Vol. 61, Springer-Verlag, Berlin, Heidelberg, New York, 1978.
- [Bjorner 80] Bjorner, D., *Reference Manual for the Meta-Language in Toward a Formal Description of Ada*, Lecture Notes in Computer Science, Vol. 98, Springer-Verlag, Berlin, Heidelberg, New York, 1980.
- [Chedghey 87] Chedghey, C., Kearney, S., and Kugler, H.-J., *Using VDM in an Object-Oriented Development Method for Ada Software*, in *VDM - A formal Method at Work*, proceedings of the VDM-Europe Symposium 1987, Lecture Notes in Computer Science, Springer-Verlag, 1987.
- [Jones 86] Jones, C.B., *Systematic Software Development Using VDM*, Prentice-Hall, 1986.
- [Mac an Airchinnigh 87] Mac an Airchinnigh, M., Burns, A., and Chedghey, C., *Reusable Units - Construction Methods and Measure*, in *Ada components: libraries and tools*, Proceedings of the Ada-Europe International Conference, Stockholm, May 1987, Cambridge University Press, Cambridge 1987.

- [Organic 82] Organick, E.I., *A Programmers View of the Intel 432 System*, McGraw-Hill, New York, 1982.
- [Pfaff 83] Pfaff, G.E., (Ed.) *User Interface Management Systems*, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1983.
- [Robson 81] Robson, D., *Object-oriented Software Systems*, Byte, August 1981.

Project No. 1062

A C C O R D

A STEP TOWARD INTEGRATED COMPUTER AIDED ANALYSIS ENVIRONMENTS

J.P. PATUREAU

ESPRIT PROJECT 1062

Athens School of Economics and Business Sciences

BERTIN & Cie

GEC Research Ltd

Nederlandse Philips Bedrijven VB

Société Générale de Techniques et d'Etudes

Trinity College DUBLIN

Universita Degli Studi di Genova

Vector Fields Ltd

1. BACKGROUND AND MOTIVATIONS

Much of the commercial future of any industrial product is deeply rooted in the design process when most of the options are specified. Manufacturing, while having a definite impact on product quality and cost, cannot amend inadequate performance, poor built-in reliability or high resulting maintenance cost for example, which all drastically affect the commercial success of a product and thus the reputation of a company. More comprehensive and efficient design of advanced engineering products is therefore one of the keys to improving the competitiveness of European industries.

For many, design (and particularly CAD) is associated solely with defining and arranging shapes and volumes or specifying relative positions of parts. Whereas this is indeed the final manifestation of design, eventually leading to manufacturing, it is only the result of an elaborate process of successive optimizations and trade-offs within a specified framework. Innovative design can be considered as drawing from a distillate of past experience and an awareness of the requirements both at the product level (specifications) and at the company level (economic and industrial constraints). However, the level of product sophistication is becoming such that designers skills will be insufficient to carry out product optimization unaided, as is already the case for VLSI design for example and also for many high technology pieces of equipment (space, military or nuclear). Mathematical modelling will provide a major key to further progress contingent upon the ability to accelerate computational processes by at least one or even two orders of magnitude (even more in the future) thus putting an answer within reach in minutes instead of hours. New computer architectures are emerging which will soon be available to significant fractions of the designer community. Some have already reached a

fairly mature technical and commercial status such as vector add-on processors. Others, relying on true parallel processing (hypercube or transputer types for example) are being extensively developed and will become part of the design toolbox in the near future. Getting the designer community ready to make full use of parallel processing is therefore an essential block in building up the enhanced Computer Aided Design environments of the 1990's.

A design team usually operates in a complex company framework (involving company know-how, strategy, commercial status, manufacturing facilities, etc...) which acts both as a source of constraints and as a source of information and data highly relevant to the design process. Because this process involves a complex range of interdependent activities which are inevitably carried out in parallel (optimizing shapes and volumes, checking on performance, checking on reliability and quality, evaluating costs, ... to name but a few), one of the challenges of CAD development appears to be in the true integration of the tools and methods used in the various design activities, around a common and unique base of data, facts and general information and through a coherent and uniform user interface. This is altogether a formidable task as it touches upon many aspects of industrial organization and practice, and yet it is clearly a key element of CIM.

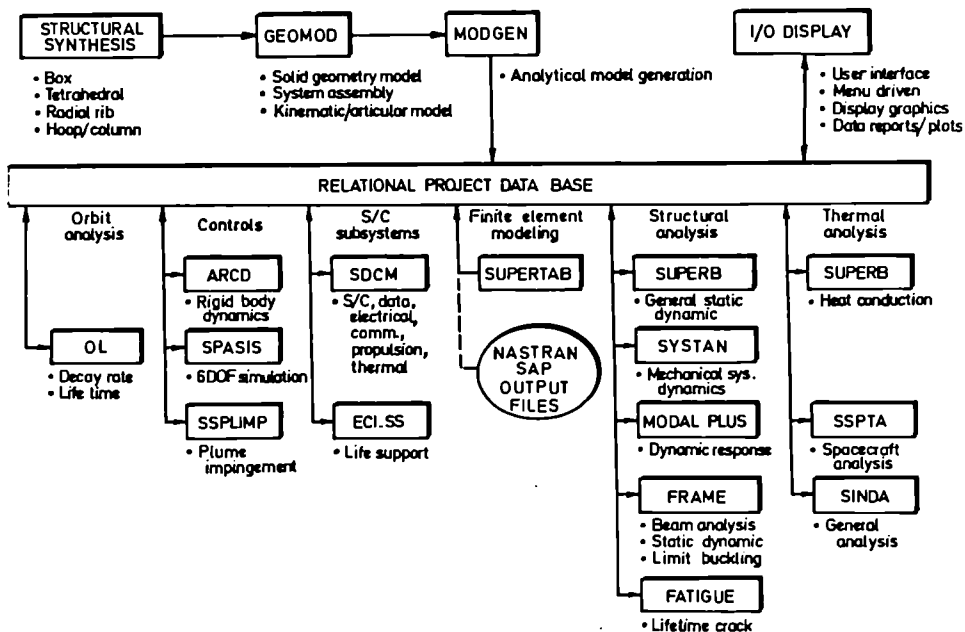


Fig 1 - NASA IDEAS / SDRC I-DEAS INTEGRATED CAPABILITY

The motivation for ACCORD was to recognize those basic long term needs of CAD development and to demonstrate on a test case (comprised of thermal management and reliability/cost assessment of electronics equipment) the potential of such new techniques as concurrent processing, data base management systems and knowledge representation methods as means of facilitating the designer's control over diverse engineering data, facts and information which take part in the design process. Some software vendors have already taken up the challenge to some extent. SDRC with their IDEAS suite are a good example for the design of space systems (see Figure 1 for a synoptic of their package). Yet, it is in the encapsulation of design expertise where the ACCORD facility will provide an important advance over projected CAD vendor products.

2. OBJECTIVES OF ACCORD

The kind of novel CAD software environment described earlier is ideally meant to be multidisciplinary (any advanced product will involve diverse fields of engineering : fluid mechanics, electrical, stress analysis, dynamic analysis, heat transfer, reliability/availability evaluation, life cycle costing estimation, and possibly many more) and multi user-type (expert in one discipline, product engineer or project manager as some kind of a generalist user, or even less experienced designers). Obviously, the task is overwhelming and one should rather look, in a given project, for a mechanism (demonstrated on a test case) by which to initiate the development of such software environments. This is what ACCORD is about.

ACCORD will be a software facility to be used, on workstation-type terminals, as a complement to existing CAD products. In order to maximise the benefits to and the impact on the designer community at large, ACCORD will be comprised of two individual and independent software packages.

- a toolkit of generic software modules for concurrent processing of the computationally intensive tasks in mathematical simulations. The toolkit is directed at the simulation tool developer who will be in a position to select whatever routines (for a given hardware) make up his application and who will also see his job eased and accelerated by construction facilities provided in the toolkit. This toolkit is expected to have a considerable impact on all segments of the engineering community which, in some way or another, develop or adapt finite element, finite difference or finite volume methods, as well as those dealing with stability and graph analysis methods.
- a software environment aimed at designers of electronic equipment in high technology products in the military, space and avionics industrial sectors. This package will result from the integration, around a common data base, and through a common user interface, of :
 - . a catalogue of computerized methods in thermal management based partially on artificial intelligence for design expertise and making use of a range of analysis techniques (from analytical to complex finite difference models) as is usually done during the various phases of the design process
 - . a reliability and cost assurance facility including a software suite for the evaluation of life cycle costs of electronic equipment

Initially the two packages will be independently demonstrated. However, as soon as possible, the more intensely compute bound modules in thermal management and reliability analysis will be replaced by the appropriate toolkit routines.

Thermal management and reliability/cost estimation are but a few of the engineering disciplines involved in the design of electronic equipment. In addition to being quite useful in many instances (increasing the early impact of reliability studies for example), these two disciplines act as a vehicle for exploring and demonstrating the benefit that can be drawn from data base and AI techniques in managing engineering data as well as the limits thereof. Naturally, if this approach appears successful enough, further extensions of ACCORD would need to address and incorporate other disciplines such as mechanical and electrical engineering.

3. LIBRARY OF CONCURRENT MODULES

The ACCORD toolkit is a subroutine library for engineering applications available to code builders using Vector and/or concurrent architectures.

In any design analysis system there are a large number of common well defined numerical procedures, eg the solution of algebraic equations, numerical quadrature, etc.

These procedures will be collected into a structured library of software with the purpose of providing the CAD system developer with the basic tools to build an applications package or to improve the performance of an existing package.

An outline example of a typical software and hardware environment is shown in figure 2.

As a toolkit for application code developers, the library will be similar to existing libraries for sequential machines, eg like the NAG library. It will therefore contain a number of basic routines with clearly defined input and output parameters, error handling mechanisms and comprehensive documentation for the user. The major difference, of course, comes in the fact that each routine will have been optimized for running with maximum efficiency on a given class or even a given type of concurrent hardware.

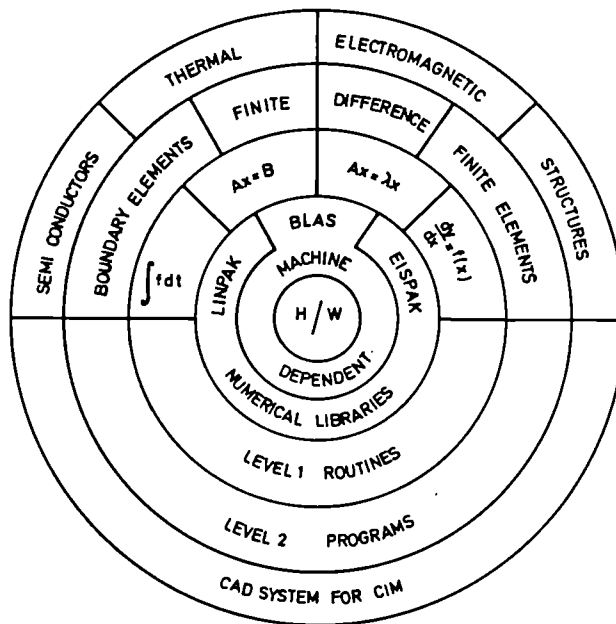


Fig 2 - ACCORD TOOLKIT ENVIRONMENT

3.1. Hardware

The first difficulty in such an endeavour is clearly to select the hardware that will be considered. The number of architectures featuring some kind of concurrent processing is quite large indeed and ranges from very big, high priced mainframes to small, reasonably priced add-ons. Examples include the Cray family, the IBM 3090/200, the Floating Point System M 64 series, the Alliant FX series, the Computing Surface of Meiko (using transputers), the Hypercube of Intel or the Mini-Dap of Active Memory Technology. Several criteria can help in the selection of hardware for library development. To start with, the size (large main frame or small add-on) is very much linked to the kind of computer configurations that are likely to be found in industry in the near future. Although the work station concept (which favors the add-on system) is presently quite strong, the longer term trend is clearly towards heterogeneous network systems where large main frames could somehow find a place. These systems have, however, received quite a lot of attention (and money) in the past ten years, so that a large volume of software (both programming and application) is already available.

1) MATRIX, VECTOR AND SCALAR OPERATIONS	(M)
2) LINEAR SOLVERS	(S)
3) EIGENVALUE SOLVERS	(E)
4) QUADRATURE CALCULATIONS	(Q)
5) SHAPE FUNCTION CONSTRUCTION	(B)
6) COORDINATE GEOMETRY CALCULATIONS	(G)
7) NAIVE MATRIX ASSEMBLY	(A)
8) SPECIAL FUNCTION EVALUATIONS	(F)

Fig 3 - STRUCTURE OF THE LEVEL 1 COMPONENT OF
THE CONCURRENT TOOLKIT LIBRARY

For the other machines, one factor which is then discriminating on a short term basis is the availability of a suitable software development environment. This is particularly critical for massively parallel systems which are fairly new on the market.

Another down-to-earth selection parameter, namely some kind of performance measure, can be gained by implementing suitable benchmarks i.e. those representative of the kind of problems to be solved by the library routines. Three examples were chosen :

- a) dense matrix : to consider Gaussian Eliminator
- b) sparse matrix : to consider SOR + ICCG
- c) eigenvalue problem.

At this stage, two kinds of architectures have been selected on which to develop the library. Vector machines like the FPS M 64 series due to their industrial maturity (software availability, vendor support) which makes them likely to concern shortly a significant fraction of the designer community, and MIMD machines (multiple instruction, multiple data) like the Hypercube or the Computing Surface in view of their very large potential in the longer run.

MFT10S(D)	Forward Elimination for L with n diagonal band storage
MBT10S(D)	Backward Substitution for U with n diagonal band storage
MVT10S(D)	Multiplies Ax for A with n diagonal band storage
MFT20S(D)	Forward Elimination for L with irregular sparsity pattern storage
MBT20S(D)	Backward Substitution U with irregular sparsity pattern storage
MVT20S(D)	Multiplies Ax for A with irregular sparsity pattern storage
MFV10S(D)	Forward Elimination on a general square matrix
MBV10S(D)	Backward Substitution on an upper triangular matrix

Fig 4 - SOME EXAMPLES OF MATRIX OPERATION ROUTINES

3.2. Library structure

The library will contain two chapters : a vector and a concurrent chapter. The internal structure is meant to be hierarchical, with two levels to start with :

Level 1	contains basic building blocks
Level 2	contains example programs making use of the building blocks

The structure of the level 1 is shown in Figure 3. Some examples of the basic routines to be developed are shown in Figures 4 and 5. Obviously, existing vectorized or parallelized routines will be directly incorporated when necessary (examples include the BLAS routines for matrix, vector and scalar operations).

SST10S(D)	Strongly implicit symmetric solver for A with 5 diagonals Requires : MFT10S(D) MBT10S(D) MVT10S(D)
SCT10S(D)	Pre conditioning conjugate gradient symmetric solver for A with regular (n-diagonal) band storage Requires : MFT10S(D) MBT10S(D) MVT10S(D)
SCT20S(D)	Forms pre conditioning matrix for use in conjunction with SCT10S(D)
SCT30S(D)	Pre conditioning conjugate gradient symmetric solver for A with irregular sparsity pattern storage Requires : MFT20S(D) MBT20S(D) MVT20S(D)
SCT40S(D)	Forms pre conditioning matrix for use in conjunction with SCT30S(D)
SCP10S(D)	Conjugate gradient squared (CGS) iterative non-symmetric solver for A and single right-hand side B
SGV10S(D)	Gaussian elimination solver Requires : MFV10S(D) MBV10S(D)

Fig 5 - SOME EXAMPLES OF THE LINEAR SOVERS ROUTINES (Ax = B)

4. CAE PACKAGE FOR ELECTRONIC EQUIPMENT

4.1. General

The ACCORD CAE software package is meant to be used as a complement to currently existing CAD tools (printed circuit board design and routing, general purpose CAD, circuit design and testing, etc...) in a similar workstation-type of environment. More specifically in this project, it is meant to assist reliability engineers, life cycle cost analysts and thermal management engineers in their respective tasks pertaining to a common product design. The basic philosophy is not for the machine to take complete control of the design process, but rather to provide user-friendly and efficient means to the user for carrying out design optimizations and trade-off studies. Such means are nothing but a collection of the best tools, methods and experience of those, which are currently used, in their own way, by domain specialists.

The major advance proposed in ACCORD is in the integration of these tools, methods and experience in a common data base structure and in a unified access through a common user interface. The expected advantages are as follows :

- relevant data, design experience and know-how are more readily available for each domain specialist one the other hand and across domains on another hand. This should result in gaining time, avoiding errors and redundancies
- interaction between design aspects is made easier, thus allowing more coupling iterations and ending up with a better quality product
- the data base management and knowledge representation structures may provide a built-in mechanism for encapsulating design expertise, thus constantly enriching the facility at company or even user level

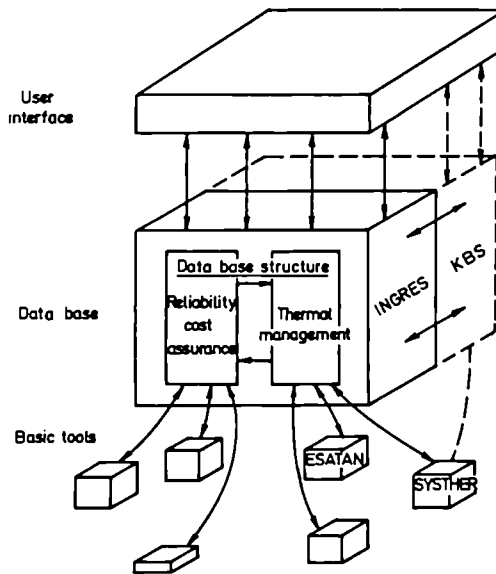


Fig 6 - STRUCTURE OF ACCORD CAE PACKAGE

The overall structure of the ACCORD CAE package is shown in Figure 6. The user interface, developed on Vax/Vms, will make full use of graphics facilities such as multiple windows (and multiple tasks), pop-up menus and mouse driven interaction. A data base structure will be built using INGRES as a relational DBMS. The base structure will contain two specific but interconnected sub-structures devoted to reliability and cost assurance and thermal management, respectively. Access to the basic computation tools will be via the data base structure as a first objective. A KBS tool, will be later incorporated at the data base level in order to enhance the capabilities of the DBMS towards managing more complex types of information (especially design expertise).

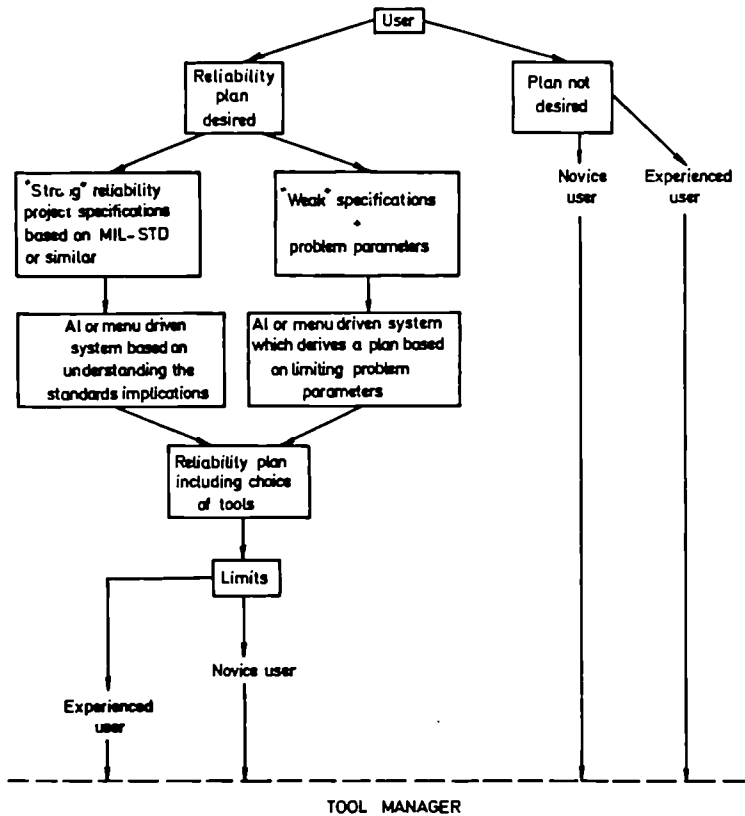


Fig 7 - STRATEGY BUILDER FOR RELIABILITY ANALYSIS

4.2. Reliability and cost assurance features

Reliability and cost analysis are techniques used by designers (project managers, design authorities, specialists) to develop product quality at optimum cost, ensuring that customer and contractual requirements are fulfilled while minimizing costs in order to meet a specified budget. Ideally, such vital considerations should accompany the design process all along its phases. The current practice is, however, often disappointing, when the above techniques are applied too late to be effective, sometimes with ill-adapted tools (with regard to the customer specifications for example) or, in some cases, even not applied at all.

Integration of these techniques in the ACCORD facility is regarded as a way towards improved practice, by providing the users with a unified, comprehensive and user-friendly environment to perform the corresponding analyses.

In the first half of the project, a basic reliability prediction tool for IC components and printed circuit boards will be integrated, which can make direct use of chip temperature calculations available from the thermal management features. This tool will be connected to large proprietary field data banks like MIL 217D, for example. In addition to the basic tool, a strategy frame work will be built as shown in Figure 7, making use of INGRES's functionalities for data management.

In the second half of the project, additional tools will be integrated to ACCORD, viz :

- . maintainability prediction
- . availability prediction
- . spares provisioning
- . Life Cycle Costing (LCC)

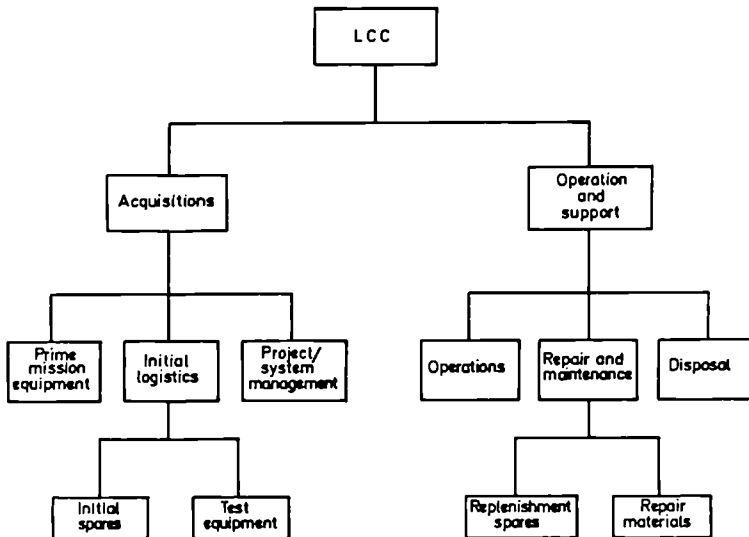


Fig 8 - COST BREAKDOWN STRUCTURE FOR LIFE CYCLE COSTING

LCC was extensively reviewed at the beginning of the project. It was found that no commercial tool (like PRICE or FAST) is really adapted to the needs of the cost analyst and, as a result, a facility for cost evaluation must be developed within ACCORD. Figure 7 presents the general cost breakdown structure that was adopted and one can find in Figure 8 the flow chart of a typical cost estimation session. Much of the work in the near future will be devoted to deriving specifications for :

- . the individual cost element models
- . the LCC tools such as : basic cost prediction, design to cost, optimization, optimum repair level analysis, sensitivity analysis, budget comparison and cost trade-offs
- . the data base structure using INGRES
- . the software architecture

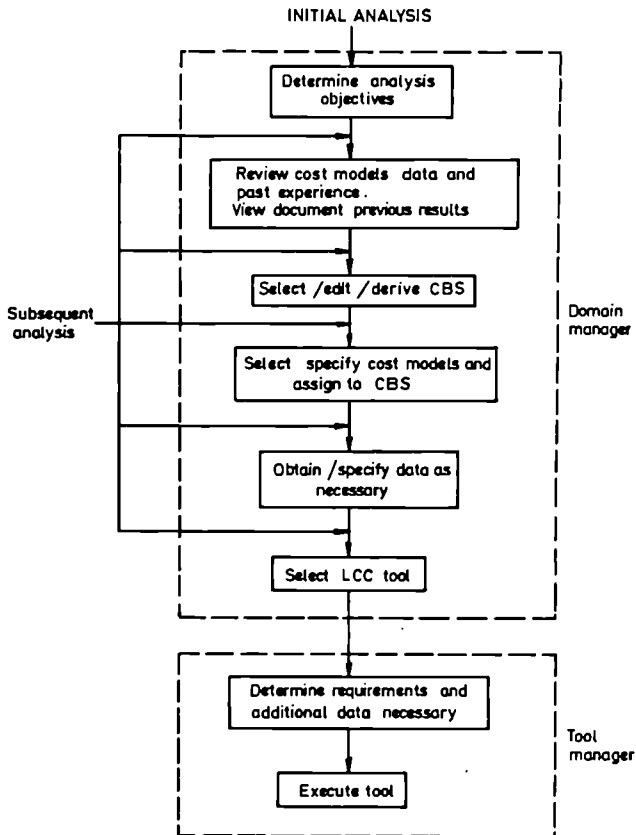


Fig 9 - LIFE CYCLE COSTING FLOWCHART

4.3. Thermal management features

Thermal management is part of the design assurance cycle for advanced space, military or avionics products. In that cycle, the thermal analyst is presented with a tentative design form for which he is asked to evaluate the thermal behavior under specified constraints.

The usual approach in thermal management starts with identifying the most important phenomena in order to "get a feeling" for the problem and also to select the solution method most appropriate to what is known of the problem and what kind of answer is sought. This is the realm of order of magnitude analyses and crude modelling. Then the problem is most often discretized to some extent whereby a complex configuration is stylized while only retaining the most significant features. Finally, the load and boundary conditions must be specified to express in a stylized but physically significant way the interaction of the model at hand with its environment. The model is then ready to be handled by usual computer methods like finite differences (which also cover lumped parameter or nodal analysis), finite elements or even, in some cases, boundary elements.

Apart from setting up the solution method (using codes like ESATAN or NASTRAN) which can be fairly straightforward if the problem is properly posed to start with, much of the preliminary work of the thermal management analyst can be reduced to a generic sequence of the following form, many times repeated :

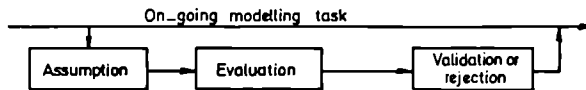


Fig 10 - BASIC ANALYSIS SEQUENCE

Such a sequence would be used for example to select or reject an important feature or phenomena in a problem, or to specify a proper boundary condition. There are two basic ingredients to those sequences :

- examples of similar problems with the way they were solved. This is inclusive of both user expertise and general know-how (company experience). Such examples can suggest ways of approaching a given problem and can also act as a concrete and relevant support for evaluation. They will further on be called technological templates,
- general heat transfer knowledge which permanently acts as a complement and a reference to, and a control of the user expertise. It often takes the form of highly stylized academic cases stressing one particular type of phenomena along with the theoretical and experimental evidence gathered by the heat transfer community, around such configurations. Those will further on be called physical templates.

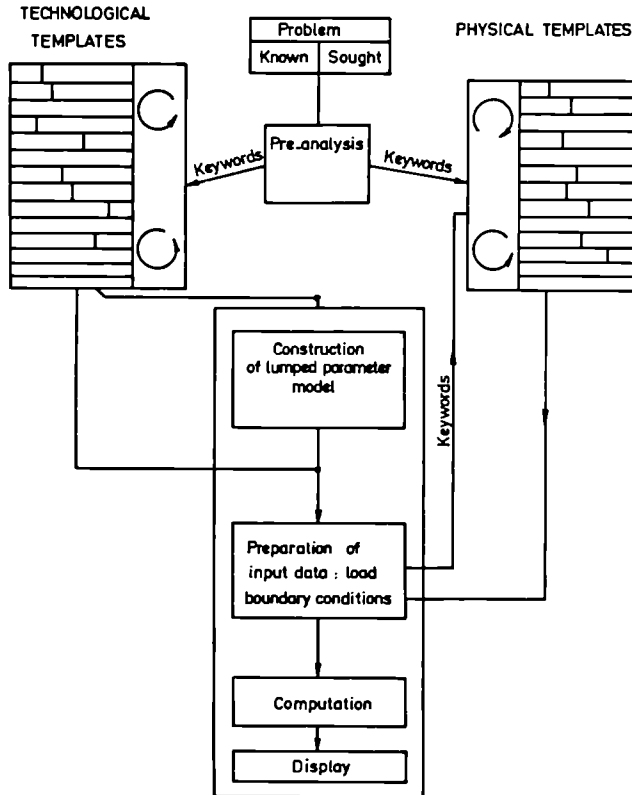


Fig 11 - STRUCTURE OF THERMAL MANAGEMENT FACILITY

The thermal management facility of ACCORD is engineered in such a way as to support the analysis sequences described above, in addition to providing computer tools for setting up, computing and exploiting thermal lumped parameter models. The structure of the facility is shown in Figure 11 where three blocks appear :

- a lumped parameter (nodal) analysis tool box constitutes the backbone of the thermal facility around two basic tools : ESATAN and SYSTHER
- an Extended Data Base (EDB) of technological templates which encapsulate expertise and examples of electronic equipment configurations
- an EDB of physical templates that contains basic, universal heat transfer problems and their solutions (see Figure 12 for an example).

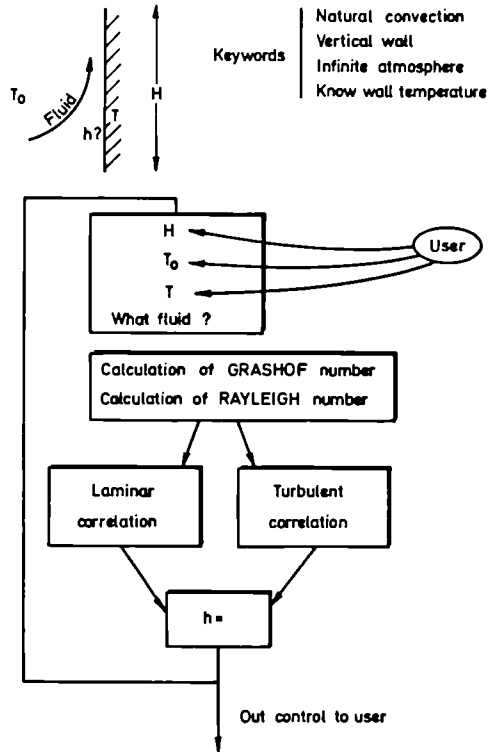


Fig 12 - EXAMPLE OF A PHYSICAL TEMPLATE

In a first phase of the project, each template is going to be characterized by a set of keywords as close as possible to specialist natural language. The keywords will be managed by a data base management system (INGRES) through which access to the facility will be provided : the user will describe the problem currently at hand and the DBMS will search for similar configurations.

At a later stage in the project, a KBS tool will be used to enrich the keyword (or extended concept of a keyword) management structure with valuable information pertaining to the template, its relevance, limitations, etc...

5. CONCLUSION AND PERSPECTIVES

ACCORD started in the Fall of 1986. Much of the early work was devoted to reviewing all the relevant aspects including :

- CAD practice and tools in industry
- design methods
- LCC practice and tools
- data base construction and management systems
- AI development tools
- hardware for both sequential and concurrent processing
- development software for concurrent processing

This work concluded on preliminary specifications for the two software prototypes that will constitute ACCORD's deliverable. The next major milestone will take place in the Fall of 1988, when the concepts which form the basis of ACCORD will be first demonstrated on a limited scale.

If this proves successful, the scope for using ACCORD in many different design areas appears quite large, indeed owing to its dual structure, viz. :

- facilities (including analysis tools and general knowhow) that are discipline-specific, say heat transfer or reliability analysis. Such facilities can be used by different types of industries and are embodied in the physical templates and general purpose analysis tools,
- facilities that are industry specific, say electronics industry as is the case here, and which are characterized by the technological templates. Such facilities must be adapted from one industry to another and can even be enriched by the user.

Finally the scope for using the concurrent module library is also very wide since the basic numerical techniques that will be addressed (finite element, finite difference or finite volume modelling) are used in just about any field of engineering.

*The constructive comment of all partners of the
ACCORD Consortium, as well as of the EEC Reviewers,
are gratefully acknowledged.*

DEVELOPMENT OF AN INTEGRATED PROCESS
AND OPERATIONS PLANNING SYSTEM
WITH THE USE OF INTERACTIVE 3D MODELLING TECHNIQUES
(ESPRIT Project 409)

Dipl.-Ing. G. Ernst

Introduction

The re-use of computer-internally stored data for down-stream connected tasks is the main emphasis of all CIM activities. This is particularly true of the re-use of engineering-design data (CAD) used for the tasks of operations scheduling (CAP) and NC programming.

The re-use of data is meant to speed up the process of preparing production records (work plans, paper tapes etc.) and to largely eliminate errors that are caused by redundant data input.

The procedure that is currently applied in industry is based on the provision of standardized interfaces (e.g. IGES or VDAFS) in the CAD systems. In the re-use of data it must be noted that it is true that the final geometry of a workpiece can be transferred, but that intermediate states of machining operations which are standard for operations scheduling and NC programming are not supported.

Therefore, it is necessary for the arriving data to be extended during the planning process by the initial states (blank data, intermediate machining states etc). Furthermore, it is necessary for the final geometry to be corrected, e.g. by the adapting of CAD data with specified tolerances to the mean size of tolerance. Moreover, it is necessary for standard geometries which appear only as text attributes in the workpiece model (e.g. recesses, undercuts, thread-runout grooves) to be incorporated into the final geometry.

It is obvious that with the procedure of data transfer that has been applied to date the goal of data integration between CAD and CAP can be attained only unsatisfactorily. Also, it must be said that, besides their actual planning functions such as, e.g. cutter-path and cutting-value determination or tool selection, the planning systems and NC programming systems must provide the same functions as CAD systems, e.g. graphic interactivity or modelling techniques. This has led to a number of redundant program developments involving high expenditure for the consumer.

The ESPRIT 409 project is intended to close this efficiency gap.

Purpose of the Project

According to its research purpose, the

DEVELOPMENT OF AN INTEGRATED PROCESS AND OPERATIONS PLANNING SYSTEM WITH THE USE OF INTERACTIVE 3D MODELLING TECHNIQUES

has set itself the goal of supporting effectively the tasks of operations scheduling and NC programming through CAD functions and CAD data. The term "CAD functions" means all the options of an efficient CAD system such as are required for the visualization, modelling or generation of geometric data.

Among others, the following tasks of planning are to be supported by CAD functions:

- determination of the blank and finished part by paging in CAD drawing archives
- engineering design of the blank
- selection of manufacturing facilities (machine, fixtures, tools) through display of graphic symbols or through display of the actual model (3D)
- preparation of the CAD model for the manufacturing facilities
- delimitation of the area to be machined
- parametrization of the machining task
- representation of the progress of machining (updating of blank) through utilization of modelling functions
- representation of cutter paths
- collision checking.

For the takeover of CAD data from systems of other vendors (e.g. via the IGES or VDAFS interface) appropriate postprocessors are made available.

Practice-proven systems and application techniques form the basis of this work. In detail, the following systems of the project partners are utilized:

- the 3D-CAD system EUCLID of MATRA Datavision, France
- the NC programming system EXAPT of the EXAPT Association for the Promotion of the EXAPT System, Germany
- service programs for file maintenance of manufacturing-facilities data and NC programs of the EXAPT system,
- system functions for computer-assisted work-plan preparation and transfer of CAD data of the EXAPT system.

In the framework of the project the abovementioned systems and/or functions are being integrated into one overall system. Through the collaboration of the Volkswagenwerk AG, Germany, as project partner and long-standing user of the a/m systems, a close orientation of the development work towards the practical requirements of industry has been achieved.

The project was started in December 1985 and is scheduled to continue for a period of 5 years. The following sections give a report on the state of work done so far.

Description of the Progress of the Work

In coordination with the project partners and the project sponsor it has been decided that the project be classified in clear individual steps, with the software-specific integration of the systems involved being initially in the foreground. The following sections of this paper will first deal with the hardware-specific structure, followed by a description of the working mode based on a case study.

Hardware-specific Structure

Currently, the following hardware peripherals are used by the separate systems:

- EUCLID:**
- tablet for menu selection and control of the graphic cursor
 - graphics screen for menu display and representation of the CAD results
 - alphanumeric terminal for display of status reports of the system
 - keyboard for the alphanumeric data input.
- EXAPT:**
- raster graphics screen for the display of graphics symbols and screen masks as well as for the representation of the results of planning and machining procedures
 - keyboard for alphanumeric data input.

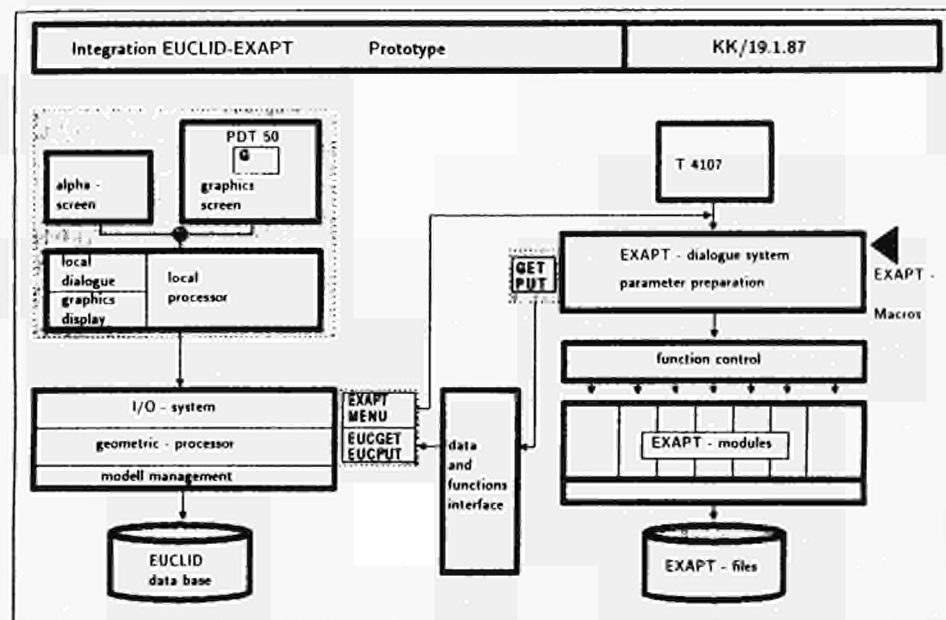


Fig. 1: Hardware-technical Structure

The goal strived for in the final extension phase of the integrated system is a unified user interface with identical input/output devices for CAD and CAP tasks. Based on the development work currently under way in the form of virtual screen-management techniques, the configuration represented in fig. 1 has been chosen as an interim solution. In this configuration, the screen of the type 4107 takes over the guidance of the user through the planning tasks as well as the representation of the alphanumeric results of planning. The PDT 50 screen takes over the manipulation of CAD data and the representation of the graphic results of planning.

Working Mode of the System

For better explanation of the functionalities and application techniques currently in the process of development, the working mode of the integrated system is explained in the following paragraphs from the point of view of application. This is done with the aid of a case study. Figures 2 and 3 show excerpts from the user interface.

First, the CAD system is started in the normal manner. By entering the identification number of the model the workpiece model (finished part) is activated by the CAD data bank and represented on the PDT 50 unit. Then, the CAD link module that has previously been integrated into the CAD system as an application module is started. Now, the selection menu for the CAP environment will appear on the T 4107 unit. With this, the initialization prerequisites for both systems have been created.

Via screen masks on the T 4107 unit the user is being guided through the task of planning. First, he is asked to prespecify the blank. For this purpose he can alternatively either prespecify the identity number of the blank of the CAD model or he can design the blank with the aid of CAD functions. In the first case, the blank model is being directly activated by the link module from the CAD data base. The user places the blank in graphic dialogue in relation to the finished part.

In the latter case, the user leaves the environment of the link module, proceeds to the CAD environment and designs – with the functions of CAD – the blank around the finished part.

Finally, the user initiates a Boolean "AND" operation between the blank and the finished part. The result thus obtained is the volume to be machined which is displayed in another colour on the PDT-screen.

In the subsequent steps, this volume to be machined is divided up into separate machining segments. This is done under consideration of production-specific aspects which are based on the knowledge of the work planner and the available manufacturing facilities of the company (machines, fixtures, tools etc.). The following sections will describe the procedure followed by the planner in parametrizing a partial operation.

First, the data of the desired shape are asked for. For this purpose a screen mask which contains the characteristic data will appear on the T 4707 screen, e.g. for a hole:

- diameter
- starting point
- direction vector
- length.

These data can be determined by selecting the appropriate screen window (which displays

the latest data content of a parameter) via CAD functions, e.g. determination of distance or direct takeover of a datum. If the CAD data structure contains more complex data contents regarding the shape, then all data can be activated by picking on the graphic screen. The elements selected from the CAD system are marked in colour; the data thus obtained for a shape element are displayed alphanumerically in the manner previously described on the T 4107 screen. These data can then be corrected by means of additional functions from the CAP environment, e.g. in the easiest case via editing techniques or, in a more complex case, via tolerance-calculation algorithms. Thus, the *first important step* of the transfer of geometric data from CAD regarding the segment to be machined is completed ("what is to be machined?").

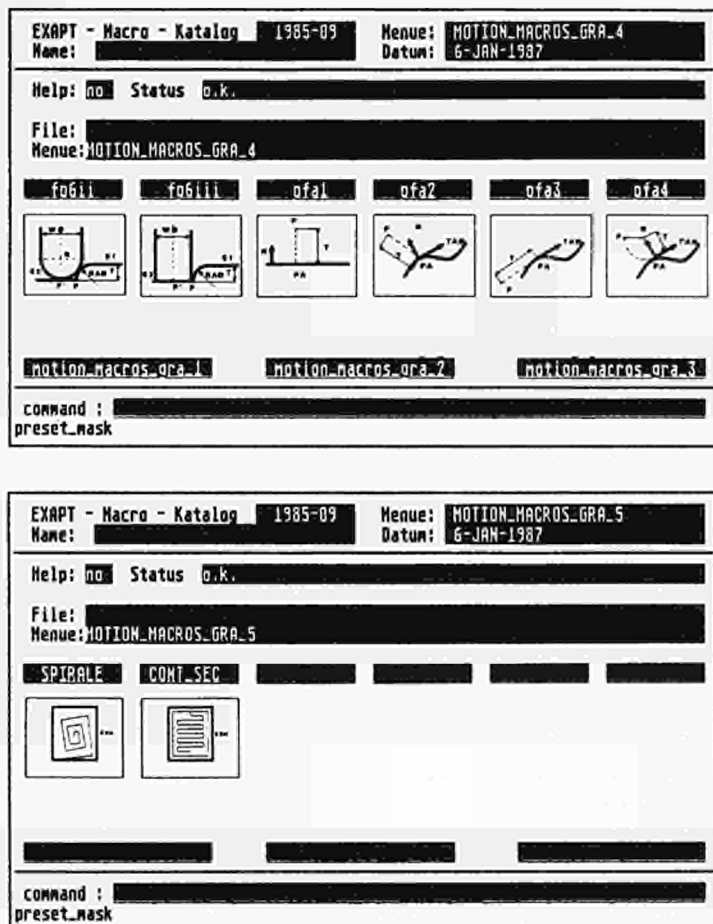


Fig. 2: Excerpts from the User Interface (Selection of MACROS)

The *second step* deals with the planning of the machining method ("how is the machining to take place?"). Normally, there are several different machining methods for one segment. These methods depend on the geometric data taken over (e.g. depth of hole, final quality required, hole diameter, material etc.). For this reason the existing machining methods are

offered on the T 4107 screen in the form of graphic symbols with textual support (fig. 2). From this array of solutions the user will choose one solution.

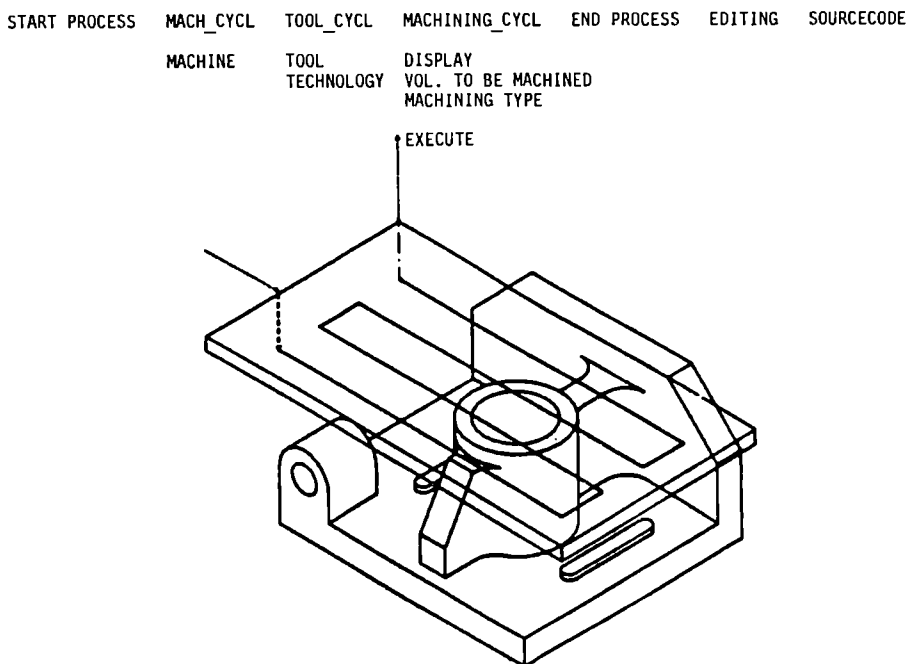


Fig. 3: Excerpts from the CAD User Interface

To the machining method, in turn, different manufacturing resources are allocated such as clamping devices, clamping methods, tools, etc. ("by what means is the machining to be done?"). In addition, it is necessary for the machining conditions to be defined, e.g. prespecification of cutting values, utilization of cooling agents, determination of correction switches. This *third step* occurs either in dialogue on the T 4107 screen or via CAP/NC algorithms.

Finally, in the *fourth step*, the cutter paths and technological data can be determined with the aid of CAP functions.

In the *fifth step*, the results of planning are displayed within the CAD environment. In addition, the data of the just generated shape element are used to update the blank in order to provide the user with the current intermediate state of the machining process, enabling him to proceed with the next step.

This procedure is continued until the workpiece has reached the required final shape.

This is then followed by the overall viewing and the judging and assessment of the results of planning. In this connection, it is necessary for modifications, if any, of the individual tasks, of the machining sequence and of simultaneous actions, if any, to be taken into consideration. However, these actions will not be dealt with in detail at this point.

Some important functionalities mentioned earlier have already been achieved in a prototype version. The elements of integration which are of particular importance for the set tasks will be dealt with in more detail in the following sections, taking particularly into account the incorporation of the user's know-how, the system structure for the division of labour between user and system supplier, and the data transfer between the systems involved.

Incorporation of the User's Know-how

The case study described above makes it clear that an important aspect of the construction of an integrated CAD/CAP/NC system is the flexible configuration of the system components. The planning sequences and planning data must be of such a nature that they can be adapted to the requirements of the company (e.g. range of products to be machined, manufacturing structures, work-paper formats, existing files for manufacturing resources and facilities, planning data for determination of allowed (standard) time). For this purpose, techniques are required which enable the user to configure the system to suit his requirements. Such techniques, which on the user level include screen masks, graphic symbols (ICONS), help texts etc., will from now on be referred to as so-called "user elements." From the point of view of system configuration it is understood that "user elements" serve not only to define the user interface referred to above, but also to formulate system procedures in a problem-oriented language.

The EXAPT language and EXAPT MACRO technique have proven their value for the definition of user elements for production planning and NC programming. By these means the system administrator is in a position to store the expert know-how of the company. This technique will be used consistently within the framework of the project work. Because the current task of a CAD integration goes beyond the task of NC programming, the EXAPT language is being extended within the framework of this project by the following functions:

- CAD takeover and transfer techniques
- interaction techniques e.g. for the definition and activation of screen masks, menus or graphic symbols
- text-processing techniques
- relational data-bank techniques.

Structure of the Integrated EXAPT-EUCLID System

With the application of user elements in an (extended) EXAPT-MACRO technique the integrated system is represented on the following 3 levels:

- system level
- (extended) EXAPT interpreter
- application level.

The *system level* contains all functions of the systems involved, e.g. for visualization, processing of 2D/3D geometry, technology planning, EXAPT and EUCLID data banks etc. In addition, these functionalities are being extended within the framework of the project work (e.g. functions for dialogue guidance, text processing, takeover and transfer of CAD data). The EUCLID functionalities are activated centrally by means of a link module which is integrated in the EUCLID system as an application module (see also fig. 1).

The *interpreter level* serves to activate the individual functionalities. For this purpose, the user elements are activated and interpreted by data banks. The results obtained from CAD takeover actions and dialogue functions are reconverted to the EXAPT language level in order to permit corrections. With this technique a uniform readable data bus between the system functions is obtained. The result data obtained from planning tasks are stored in an operational data base.

The *application level* in a company is operated by the work planner, calculator or NC programmer, using the graphic-interactive techniques of the EUCLID system and the screen-mask techniques of the EXAPT system.

This concept of levels is meant to serve different groups of people by facilitating preparation and application of the system:

- system developers (CAP/CAD vendor's staff) for further development, maintenance and care of the system functions
- application programmers (user-company's staff) for system configuration in the form of user elements
- consumer.

It is intended within the framework of the project work to prepare standard user elements for the most frequently occurring tasks of the CAP-NC area.

Data Exchange between EUCLID and EXAPT

Besides the activation of functions, the data utilization of CAD data for down-stream connected CAP tasks, for instance, plays an important role within the framework of a CAD/CAP system integration. Although it is possible through specific activation of the relevant CAD functions to reduce the quantity of data to be transferred, it is nevertheless necessary to make data-structure analyses, particularly in the case of 3D CAD model data.

The reason for this is that CAD and CAP models have been computer-internally realized as objects in the form of trees and networks. This applies to both EUCLID and EXAPT. It is important for the data transfer that the object structure remains recognizable for the inquiring system. Furthermore, it is necessary for attributes, relationships, data and objects linked to the structure to be transferable.

For this purpose appropriate structure-coding and decoding functions have been developed which take care of the uniform transfer of all object data. These functions have been integrated into the link module.

For data transfer between EUCLID and EXAPT and vice versa a readable interface has been developed in EXAPT notation. This technique permits, on the one hand, data corrections by the user and, on the other, flexible configuration of the user elements (MACROs, screen masks etc.) for company-specific re-use.

The data structure of the transfer interface is built up according to different types. All transfer objects are always distinguished according to organizational and descriptive data. Organizational data include, for instance, identifying data (object name and/or address), classifying data (type of object, object attributes), information about relationships to other objects (owner, member) or the type of data description (data type, format, record length). The object-descriptive data include for each object element the type of data element (e.g.

coordinate, datum, etc.), the data format and the actual value. With firmly defined object structures, for instance for the takeover of a point, fixed structures are planned, e.g. with descriptive data.

Utilization of "type-classified" data structures offers the advantage of a specific takeover of data, for instance for simple takeover of an object address of CAD (e.g. a volume to be isolated). At the same time this offers the possibility of transferring data which – although existing in the sending system – cannot yet be processed in the receiving system. Furthermore, this allows the quick detection of missing data, e.g. attributes of geometry in the receiving system, and the correction of the situation by additional user entries.

Attributization of the CAD Model Data

The presented case study as well as the techniques used for the CAD/CAP data transfer make it clear that particularly important efficiency improvements in system integration can be achieved when the CAP model data and the data underlying the manufacturing task are identical. This is particularly the case when in the CAD data base the shape of the section to be machined, including all technological data such as surface quality, tolerances, etc., are attributively linked with the geometry which can be picked on the screen. By picking on the screen the shape data are transmitted via the transfer interface to the CAP user element which is automatically parametrized by the CAP data. In this case, all that needs to be done is to enter the data of the manufacturing method (e.g. tool, cutting values etc.) or to automatically activate shape-related user elements which carry out the manufacturing method. Principally, the technique of allocation of shapes to manufacturing methods which are represented in fig. 4 are offered.

- **Type 1: CAD contains the shape data; method exists in NC**
 - identify the shape in CAD
 - allocate the method to the shape
 - select method
 - parametrize the method
 - define the sequence of operations
 - optimize the sequence of operations
- **Type 2: CAD does not contain the shape data; method exists in NC**
 - select method
 - define the shape by means of CAD/NC facilities
 - parametrize the method
 - define the sequence of operations
 - optimize the sequence of operations
- **Type 3: CAD does not contain the shape data; method does not exist in NC**
 - single-machining operation using the geometric tools available in CAD and
 - in technological EXAPT facilities

Fig. 4: Types of Application of Allocation of Methods to Shapes

One of the main emphases of further developments is the creation of basic techniques for attributing the CAD geometry. Significant extensions of the CAD data structure are necessary here. Another main emphasis is the further development of CAP user elements on the basis of extensive machining analyses of the machining procedures turning, drilling, milling and measuring. In these analyses all stages of planning, including those with a high cost consideration, are taken into account. Further emphases of the work are:

- extension of the link module
- improvement of the user interface.

Discussions with potential user firms have led to the conclusion that this goal and the solutions being developed both correspond to the users' requirements. Significant economic and technical advantages can be expected particularly in the case of complex tools and small lot sizes.

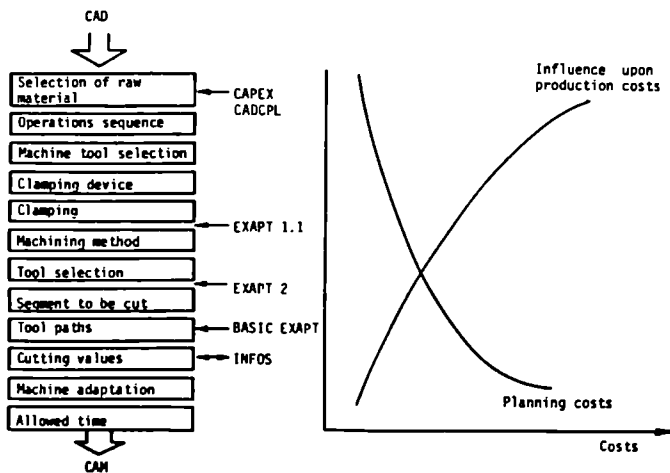


Fig. 5: Steps of Planning: Cost Causer / Cost Responsibility

Expert Supervisory Control of a Flexible Manufacturing System

H. Heinz-Fischer*, R. Los⁺, H. Poels⁺, C. Pohl*, J. Ringel*, H. Stienen⁺

⁺ Faculteit voor Technische Wiskunde en Informatica, Technische Universiteit Delft,
Julianalaan 132, NL-2628 BL Delft

* Krupp Atlas Datensysteme GmbH, Altendorferstraße 104, D-4300 Essen

A prototype of an expert supervisory system to control the work flow through a flexible manufacturing system (FMS) has been developed. The software package contains three cooperating programs: a real time expert system, a dynamic scheduler and an FMS-controller-simulator as a substitute for a real FMS. The expert supervisory system can perform scheduling of orders, prediction of work flow for the created schedule and simulation of the execution of orders in real time. During operation a dynamic change of the scheduling strategy is possible. This can be done autonomously by the expert supervisory system or manually by the cell operator. Main module of the package is the expert system that supervises the predictive scheduler, the FMS-controller-simulator and some additional activities such as the selective display of information relevant to the operator.

The objective of this investigation is to demonstrate the feasibility of both expert systems and dynamic schedulers in a real time environment.

Keywords: Scheduling, Decision Support, Expert System, Real-time Processing.

1. OBJECTIVES AND BACKGROUND

The aim of the Esprit Project 809 "Advanced Control Systems and Concepts for Small Batch Manufacturing" is to improve the machining of mechanical parts in small to medium batches. As a part of this project, in Work Package 2 (WP2) the consortium members TU Delft and Krupp Atlas Datensysteme have developed a prototype of a dynamic scheduler supervised by a real time expert system to control the work flow through a flexible manufacturing system (FMS).

Since in a flexible manufacturing environment all operations are highly interrelated due to the limited availability of shared resources such as buffers, pallets and tools, scheduling is more complex and more critical in an FMS than in traditional job shops. Current schedulers operate remotely as a separate subsystem. They do not have any knowledge of the actual status of the FMS. As many elements may affect the flow of work through the cell, conventional scheduling strategies do not react quickly enough to achieve a reasonable workload. By introducing production control into the flexible manufacturing cell itself, the supervisory system will be kept fully aware of any deviations from the predicted production progress, and decisions can be made quickly in order to make optimal use of the available resources.

As a consequence, real time reordering of already scheduled operations should be possible to some extent. Due to the rapidly changing environment in which a dynamic scheduler has to operate, it is reasonable to expect that an expert system will be capable of supervising the scheduling task, as in such a system the control knowledge is encoded in a way that adaptation is easily done. As rule based expert system shells are now mature products, their use in industrial applications to implement a scheduling and dispatching system can be envisaged. But as most shells focus on diagnosis and classification, embedding a rule-based expert system in a supervisory system will require additional features that are not provided by available expert system shells.

Further, new scheduling strategies need to be explored, as mathematical methods from operations research suffer from the problem of combinatorial explosion because of the number of schedules that have to be analyzed to find an optimal or near optimal production plan.

A major objective of Project 809 as a whole is to provide a prototype of a supervisory system to control the work flow through an *existing* manufacturing system. As a first experiment within WP 2, a restricted prototype consisting of a real time expert system and a dynamic scheduler together with a simplified data model containing all relevant data on work orders and machining facilities derived from a representative factory has been built to demonstrate the feasibility of this approach. Thus the possibility of an expert system assisting on cell level is examined in this paper.

2. INTRODUCTION

Job-shop scheduling suffers from the combinatorial explosion of production plans that have to be generated and analyzed to determine the best one. Moreover, various conflicting production goals are candidates for optimization, thereby making scheduling strongly dependant on individual observations of the system, and highly sensitive to subjective judgement.

On the shop floor, and also within most off-line master scheduling systems, fixed scheduling strategies are used, from which it is known that on the average they behave reasonably well under idealized conditions, that show little resemblance to the real situation. Most of these techniques utilize simple rules for associating priorities to individual jobs based on such characteristics as the job's processing time or its slack time. The resulting production plan deviates from the desired optimum to such an extent that at first sight human operators are tempted to suggest their own solutions.

Single scheduling rules consider only a small fraction of feasible production plans, but generate in most cases solutions that are not too far from the desired optimum. However, single rules occasionally tend to generate schedules close to the worst case upper bound. Assuming that not all scheduling rules show this degradation of performance under the same circumstances, then trying different rules should eliminate the worst cases [5]. So, giving the supervisory system the opportunity to compare the outcomes of several alternative scheduling rules should lead to a better performance of the flexible manufacturing system. In order to limit the computational overhead, rules based on human experience are applied to select appropriate scheduling strategies.

2.1 Review of Related Research

The work of M. Fox et.al. [4] on constraint-directed search for job-shop scheduling has made this problem a well-accepted domain for the application of artificial intelligence techniques, although it turns out that the results of their investigation are difficult to apply to other locations, as flexible manufacturing systems possess highly different capabilities with respect to cell configuration (duplicate machines, transportation), arrival of orders (master scheduling) and decision criteria. Meanwhile, several experimental systems have been reported [2,3,6].

Scheduling rules can be classified into three categories [7]:

- *Loading rules* to determine the best job for a single station using only information concerning that machine. Loading rules are derived from or supported by queuing-theory experiments, where random arrival of orders at a server and infinite buffer capacities are frequently assumed. Arrival rate, operation duration and the number of uncompleted operations are most widely used as a decision criterion.

- *Sequencing rules*, aimed to constrain the search for a complete factory schedule to those areas of the solution space, where most likely the optimum will be. As complete enumeration of feasible schedules is impractical, sequencing rules are often derived from Monte-Carlo simulations. Most of the time, a combination of simple loading rules is proposed to meet the overall objectives [1,10].

- *Heuristic rules*, not founded by a theoretical model, as most manufacturing systems are subject to so many random disturbances of the work flow that the advantage of operating from an optimal, but fixed schedule is precluded. Heuristic rules incorporate a consideration of the global optimization policy and take account of interaction between individual servers and tasks. Frequently, heuristic rules govern the assignment of priorities to each job. What is the best priority rule in a particular case, such as giving highest priority to jobs that pass the most heavily loaded machine, or to complex jobs that most likely will show defects that will need some rework, will depend heavily on performance measures as well as on actual system parameters.

We found that most capacity losses are due to unpredicted events, and we decided therefore to place the major emphasis on proper exception handling, with minor attention to optimal scheduling under regular conditions. Therefore, most rules should only come into play in emergency cases. For instance, when a machine breaks down, they would be used to decide on the trade-off between staying on that machine and waiting against moving the job to a second machine which involves extra overhead as the transportation of special tools and the generation of new NC-programs.

Within the framework of WP2 no assessment of the quality of certain rules will be carried out. Only an attempt is made to decide whether it is possible to give system configurators the opportunity to formulate their preferred strategy by means of rules that are handled in real time by an expert system shell.

3. REQUIREMENTS

It is assumed that a master planning and scheduling system makes a preferred routing plan at distinct intervals for a number of orders that then have to be processed during the succeeding few shifts. So it is assumed that the FMS maintains its own list of orders, which is delivered to the cell periodically. The problem now is how to dispatch these orders in a sequence that meets various organizational and technological constraints while trying to optimize certain goals such as *make sure that all orders meet their due-date* or *minimize production costs*. It turns out that available mathematical methods alone are inadequate to provide an optimal schedule, as those methods can only take account of a small part of the relevant conditions. Skilled human operators are superior, and because humans often use heuristics, heuristic methods are also employed here.

The kernel of the system therefore should be made up of an expert system that should:

- monitor and control the status of the FMS
- reschedule when necessary using a Real Time Scheduler (RTS)
- adapt scheduling parameters (strategies, criteria) to meet a changing environment
- perform tests/forecasts (if desired)
- explain activities to an operator (if required)

The supervisory system should be fail save, i.e. as long as there are uncompleted jobs, the FMS should never be halted due to failure of the expert system or any other module not part of the FMS. For that reason a distributed approach of loosely coupled subsystems is preferred.

4. EXISTING TOOLS

Without the availability of existing software packages it would have never been possible to develop a supervisory control system within such a tight time scale. The packages used will be described in the following two sections.

4.1 The Expert System Delfi-2

An expert system is a computer program that is aimed to reason like experts in routine cases when they are urged to decide immediately. When an expert is asked to explain to a novice the way he comes to a conclusion, he will sum up the facts that are relevant in a given situation and the rules that can be used to derive lacking information from elementary facts.

Common to all kinds of expert system is the capability to derive more or less purposively those facts that are relevant to solve the posed problem from elementary data stored in the computer memory, or gathered from, or provided by, external sources such as sensors, databases or humans.

Rule-based expert systems are a means to come to reasonable results in a fast and pragmatic way, as they are better adapted to the human way of thinking than conventional systems. In close cooperation with industrial partners Delfi-1 and Delfi-2 were built by members of the Mathematics and Computer Science Department of the Delft University of Technology. As Delfi-2 emanated from Emycin, diagnosis and classification problems provided the majority of the applications at the beginning of this project. Delfi-2 is characterized by its uniformity: one single inference mechanism - *backward chaining*; one strategy for conflict resolution - mainly a *depth first evaluation*, but only *one instance for each context* given is possible. Powerful features for structuring large amounts of knowledge are absent. Single facts are stored in the working memory without taking account of the original context in which the parameter description is situated. As the reasoning of the system can proceed even without user influence, an explicit structure of the working memory is not necessary.

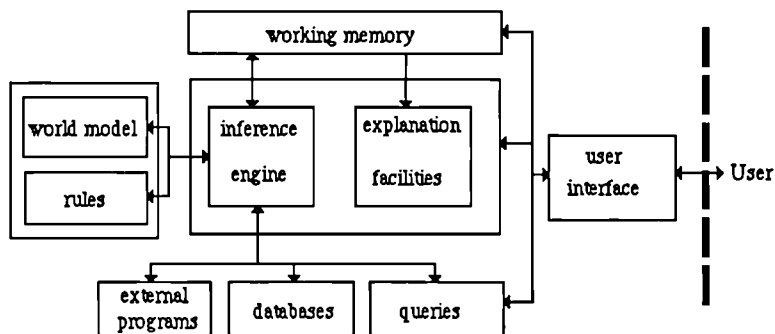


Fig. 1. The Architecture of Delfi-2

In fig. 1 the architecture of Delfi-2 is shown. An interesting feature of Delfi-2 is the *task* mechanism, made possible by the use of a general purpose language (Pascal) for implementing the expert system shell. A task is a separate program that is activated by Delfi-2. By means of a task, data can be collected or transmitted or actions like setting a signal or closing an emergency valve can be done. So, through this mechanism process monitoring and control can be realized [11].

4.2 ASIM

ASIM is a software package designed by Krupp Atlas Datensysteme for modeling discrete manufacturing processes and transport flow in factory environments by simulation techniques. Generally speaking, ASIM simulates problems related to resource allocation. ASIM models the following data entities contained in the ASIM database:

- real building blocks (e.g. stations, machine tools, buffers)
- moving building blocks (e.g. production orders and jobs)
- process plans
- product definitions (cross references part type - process plan)
- order lists (production orders defined by part type, lot size, start date/time and process plan)

ASIM is written in standard FORTRAN 77. It can be easily adapted to current operating systems. See fig. 2 for a diagram of ASIM.

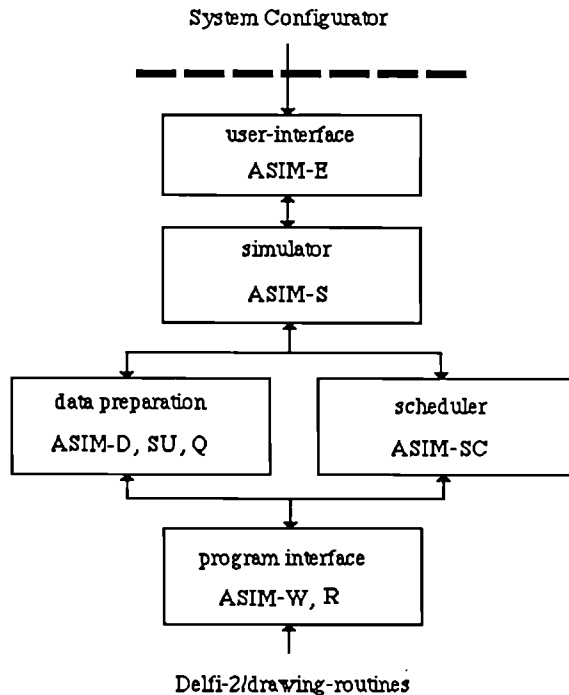


Fig. 2. Internal Structure of ASIM

ASIM proceeds by converting moving building blocks into events. That conversion occurs according to internal priorities. The events are stored in event tables. These events control the simulation progress. ASIM is capable of performing:

- pre-simulation runs (transient start-up stage)
- main simulation runs (stationary state)
- intermediate stops to exchange status data and summary results with other systems (e.g. Delfi-2)
- forecast simulations
- rescheduling according to changed strategies (as received from outside at intermediate stops)
- calculating summaries for display purposes

Each module in fig. 2 is dedicated to a special task. ASIM-E performs the model setup. Here the model is defined. ASIM-S performs the actual simulation. A presimulation is incorporated in order to get the cell into a smooth running condition. ASIM-D collects display information concerning the simulation to Delfi-2. ASIM-SU summarizes the results of the simulation and ASIM-Q adds some qualifiers to that data. These qualifiers are not important as far as simulation is concerned. ASIM-W sends data to Delfi-2 and ASIM-R receives data from Delfi-2. ASIM-SC performs the scheduling task of the system.

5. DESIGN CONSIDERATIONS

Both Delfi-2 and ASIM, being existing software packages, needed some adaptation. Concerning the expert system, Delfi-2 was modified to be able to do evaluations repeatedly without human intervention. Then in order to be able to propose not only several alternative solutions but also to make the ultimate decision, Delfi-2 needed to be extended so that it could evaluate more than one action in a rule. Further, as Delfi-2 is only capable of reasoning about single objects, while aggregate data concerning machines, orders, jobs or schedules are difficult to obtain with the aid of rules. Special procedures to collect this information are therefore coupled to the expert system as special tasks. A scheduler module and some communication modules have been added to ASIM. Modules are also provided to make status information available for display purposes.

Already early in the design phase, it was clear that the system would require a multi-processing environment. Therefore, we thought it was helpful first to establish the interfaces and synchronization between the various processes, using some simple modules, and to expand the various modules to their final versions later on. We only present the final results here.

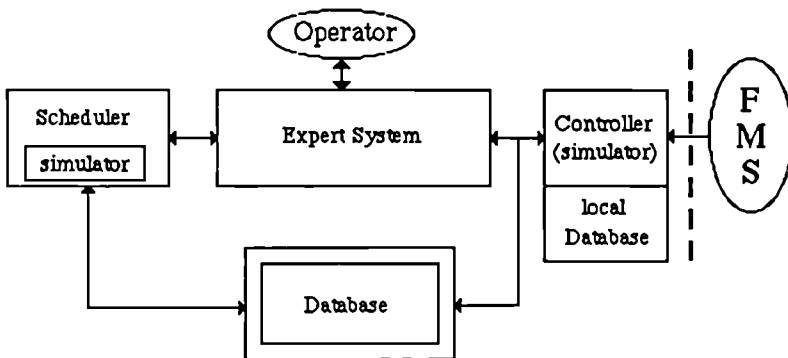


Fig. 3. Logical decomposition of the system

5.1 Logical Decomposition of the System

The software is built up of four components - a scheduler, an expert system shell, an FMS-controller-simulator and a display monitor. The scheduler however also uses a simulator to allow predictive scheduling, therefore the term simulator is reserved for the simulator in the scheduler and the term FMS-simulator for the FMS-controller-simulator. The system is shown schematically in fig. 3.

5.2 Technical Details of the Major Components

The Real Time Scheduler

The real time scheduler arranges the remaining orders left in the database and makes a production plan for them according to the parameters and strategies set by the expert system. It contains a simulator to test the generated schedule, to compute aggregate data that are typical for that sequence. If it is clear that the predicted schedule is not appropriate, the expert system will have to improve the scheduling strategy or drop some constraints (it may happen that no feasible schedule is found). The simulator used in the scheduler is the same package as used in the FMS-simulator, but for making predictions no deviations of the estimated process time and no machine breakdowns are generated. The scheduler can combine up to four strategies at a time. Primary strategies are considered as equally important and the total priority of an order is calculated as the sum of the priorities of the individual strategies. Secondary (or even tertiary) are only used in cases where several primary strategies assigned the same priority to different orders and cannot find a unique solution.

The FMS-controller-simulator

The FMS-controller-simulator emulates the behavior of a real flexible manufacturing cell. As there is no real FMS available, ASIM is used to simulate one. In the final system coming from the entire Esprit Project 809, this package will be replaced by a real FMS. Major tasks of the simulator are:

- to simulate the processing of the jobs currently in the FMS in a compressed time scale
- to generate errors and deviations that might occur in the FMS
- to send the status of the FMS each time an event occurs, which can be the start or end of the processing of a job or the rise or fall of an error flag, to the expert system and the graphical interface

The Expert System

The expert system supervises the entire process. Whenever an event is reported by the FMS-simulator, the expert system decides whether rescheduling should be done. The expert system also incorporates the user-interface to the operator. Special tasks of the expert system are:

- to monitor the status of the FMS and to decide whether rescheduling is needed
- to adapt scheduling parameters and goals to the actual status
- to explain the activities to the operator (if desired)

Monitoring the FMS enables an accurate tuning of the various scheduling parameters to a changing environment, e.g. in case of errors. If something unexpected happens, the scheduling policy may have to be changed. The expert system transmits scheduling parameters to the scheduler and poses questions to the operator in order to estimate how serious the event will be in case a machine breaks down.

The interaction between the expert system and the FMS-simulator is event driven and not based on a periodical polling process. The FMS-simulator runs continuously, so it cannot be a task (in the sense of Delfi-2 tasks) of the expert system.

Operator Facilities

The operator can basically do four things, divided in two categories:

- Ask for explanation about the current consultation. This is done using the standard explanation facility of the expert system where explanation is about inferences of the expert system. If the operator wants additional information about the status of the FMS that is not maintained by the expert system, as it is not important for the reasoning process, then some special tasks of the expert system will be activated to interpret the request and to present the required information to the operator.
- Start a forecast or a test, or enter preference data. This is done using the task mechanism of Delfi-2. In case of preference data, a request is sent to the scheduler that reschedules the orders and former data are replaced.

The operator is mainly informed about ongoing activities in a pictorial form. In fig. 4 the screen layout is outlined and Fig. 5 the GANNT-diagram is reproduced that represents the results of a forecast.

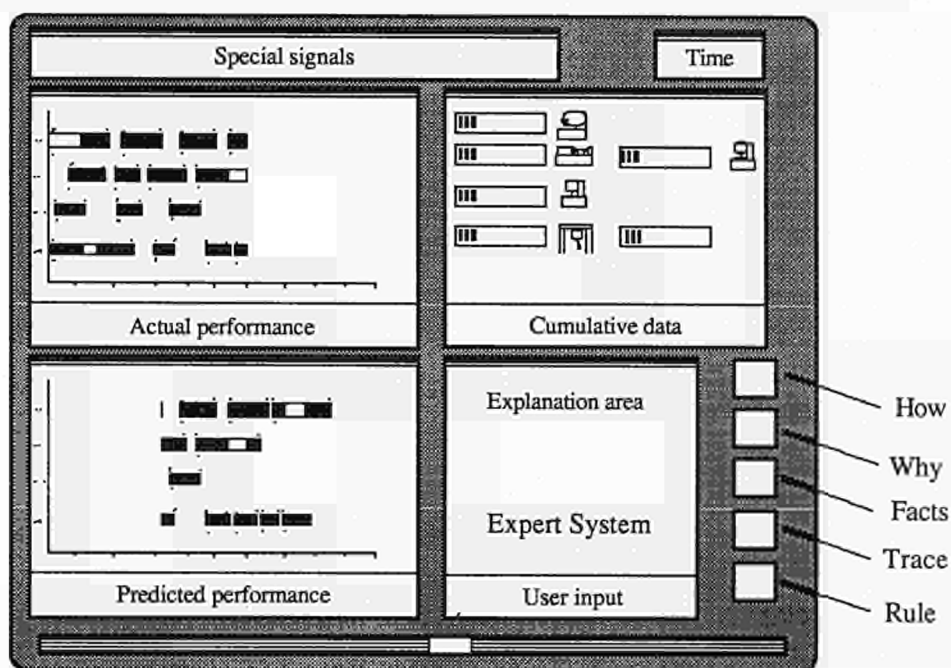


Fig 4. Screen layout of the operator console

5.3 Synchronization

Synchronization plays an important role both at machine level and at program level. At machine level problems may occur because more programs simultaneously start to transmit data. At program level, the operator may cause trouble by keeping the expert system busy at the moment when an event in the FMS-simulator requires its attention.

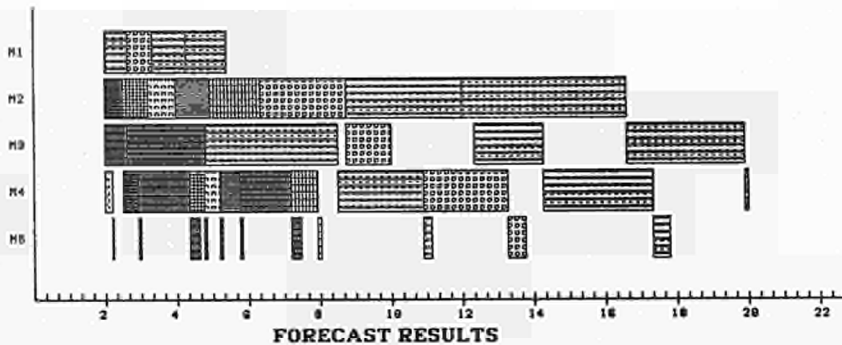


Fig 5. Example of a GANNT diagram showing the results of a forecast

Machine Level Synchronization

From the final software package one part - expert system shell, knowledge base and graphical interface - was developed by TUDelft on Apollo-computers in DOMAIN-Pascal, an dialect of Pascal. The other part - scheduler and FMS-simulator - was written by Krupp Atlas Datensysteme on an AT in standard FORTRAN 77. The Apollo DN560 and DN320 run under a combined AEGIS-UNIX operating system, the AT runs under MS-DOS 2.11 and/or 3.10.

The three programs that are permanently active, the FMS-simulator, the expert system and the FMS-display monitor, need to communicate with each other continuously. As we not wanted to modify the existing sources too much, the expert system itself communicates not directly with the FMS-simulator. Communication is handled by a mailbox, running on the Apollo. A special designed handshaking protocol enables communication between the Apollo and the IBM-AT. The mailbox acts as communication server and maintains a common database for that purpose. This database should be thought as an intermediate between the FMS-controller and the expert system. The mailbox thus serves not only as a communication device but also as a data storage device as well as an encoding and decoding program. A more detailed view of the interaction between the console driver, the expert system and the FMS-simulator is shown in fig. 6. The actual database in this figure and in fig. 2 will be substituted by a general database management system in the final version coming out of Esprit 809.

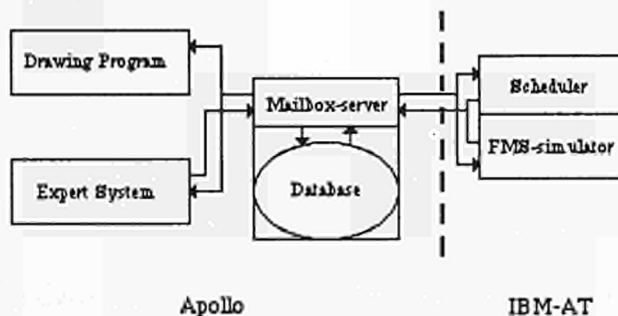


Fig. 6. Detail of connection through mailbox

The scheduler also runs on the IBM-AT and gets its data partially from the FMS-simulator and partially from the expert system through the mailbox. Communication goes like this:

- the FMS-simulator sends its complete status to the mailbox server
- the mailbox server stores the data in its database, but also sends data to the console driver
- when the expert system needs data it asks for it from the mailbox server
- when the expert system wants to activate the scheduler it sends a request to the mailbox server which then activates the scheduler and sends data needed for scheduling to the scheduler. The scheduler also needs status information from the FMS-simulator, but this information exchange is not visible to other components. (For a more precise description of the communication, see appendix 1)

The only critical point at machine level as far as synchronization is concerned, lays inside the mailbox server. Here the physical transfer of information takes place. To avoid problems we assigned a master function to the simulator ASIM. This means that each communication sequence is started by ASIM which then waits for a response from the expert system.

Program Level Synchronization

At program level, synchronization is more difficult to tackle. Problems might occur because the expert system has to monitor the FMS and at the same time has to serve the operator, when present. Certain operator actions such as asking for explanation can be done at the end of a consultation by means of the standard facilities of the expert system shell. However, operator activated forecasts or real schedulings cannot be done when a consultation has finished, but need to be incorporated in rules that are evaluated during the consultation of the expert system. The reasoning process where the decision takes place, is made non-interruptable by the operator, except at a special point at the end of the consultation, where a task checks whether the operator has pressed a key. If he did, the interrupt will be handled and a second task will enable the operator to enter the expert system, if not, the consultation will be finished and the expert system returns to the starting point. See fig. 7.

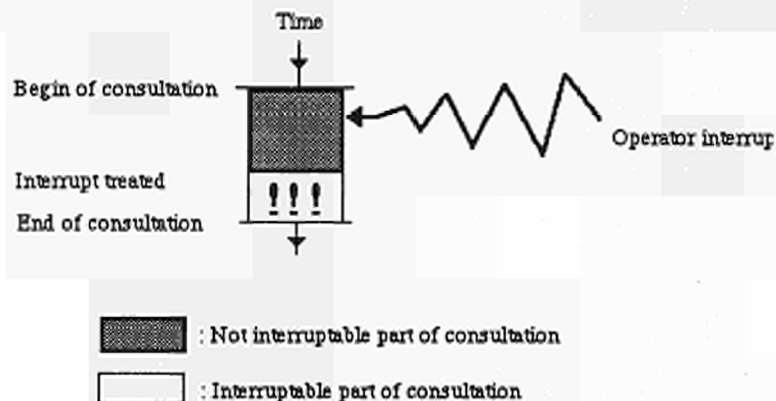


Fig. 7. Interruptible and non-interruptible periods of a consultation

5.4 Status Data of the System

Each package maintains its own status data, though it may occur that some packages access a common database. The state of the system is represented by several items grouped into different blocks:

- a set of orders that have to be processed during the succeeding shifts. Orders are subdivided into a linear sequence of jobs. Jobs do have precedence constraints. They are said to be non-preemptive, that is they cannot be interrupted until they have finished.
- a set of records containing the actual state of the FMS. Both the FMS-simulator and the expert system use this data.
- data about the state of the orders (when started, when ready, etc.)
- exception data (queue-overflows, order-delays, etc.)
- forecast results (predictive data)
- cumulative data (machine performances, average queue lengths, etc.)

Each job has its own operation and machine. Each order has a work plan attached to it describing the jobs and processing times of the jobs composing the order. This is displayed in fig. 8 where *Machine* is the machine where the corresponding job should be processed and *time* is the expected processing time for that job. When several machines are listed, alternative routes for that job are possible. Within an order precedence constraints exist in such a way that the next job in the list may only be started when the previous job of that order has finished.

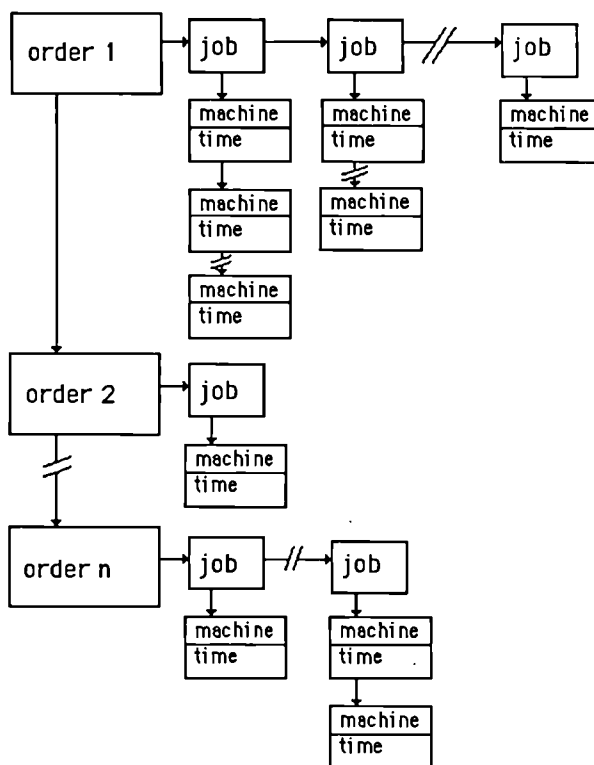


Fig. 8. Data structure containing orders and jobs

6. IMPLEMENTATION

Several aspects of a real FMS are not incorporated in the prototype. Most important discrepancies are:

- transport facilities are not taken into account
- machine capabilities are fixed, tools are provided in sufficient quantity to be always available
- setup times of machines and tools are neglected; there will be no optimization towards batch sizes (no lot splitting) or load balancing

Later versions will probably incorporate these items. It is expected that adding more details will merely be a question of tuning and will not affect the current concept.

6.1 FMS configuration and production orders

To test our system we used a simplified FMS, derived from an existing factory which should be representative [9]. In fig. 9 the (schematic) physical layout is depicted, and in fig. 10 the work flow is shown. The FMS consists of 5 stations logically arranged in a line; alternative stations are not incorporated. All orders have to pass the sawing machine, the drilling machine and the

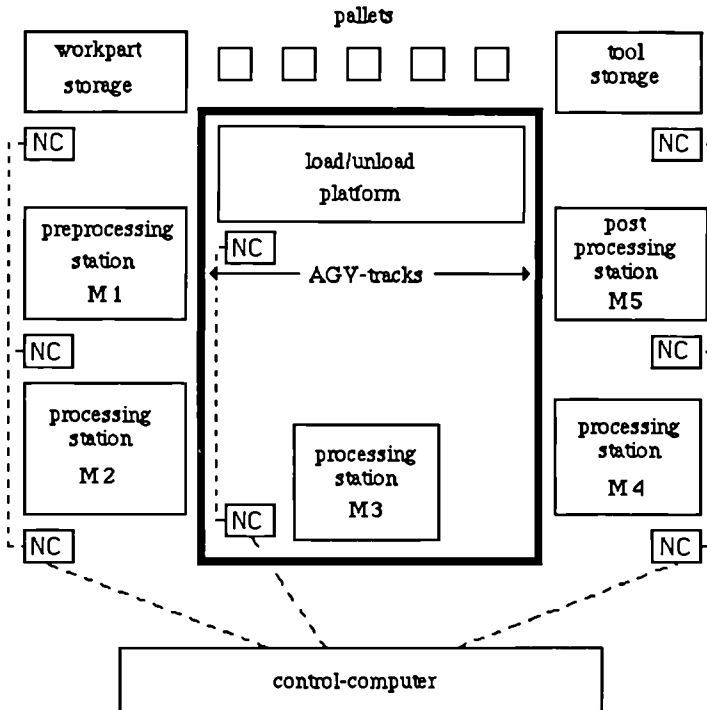


Fig. 9. Layout of the (hypothetical) FMS

measuring machine, but the lathe or the milling machine may be bypassed. The lathe, milling machine and drilling machine are said to be the essential machines. Processing times on these machines are much higher than on other machines.

Also a set of orders is composed. The product to be made is a cardan drive built up of two connecting cubes, four cardan joints, and a cardan shaft, see figure 10.

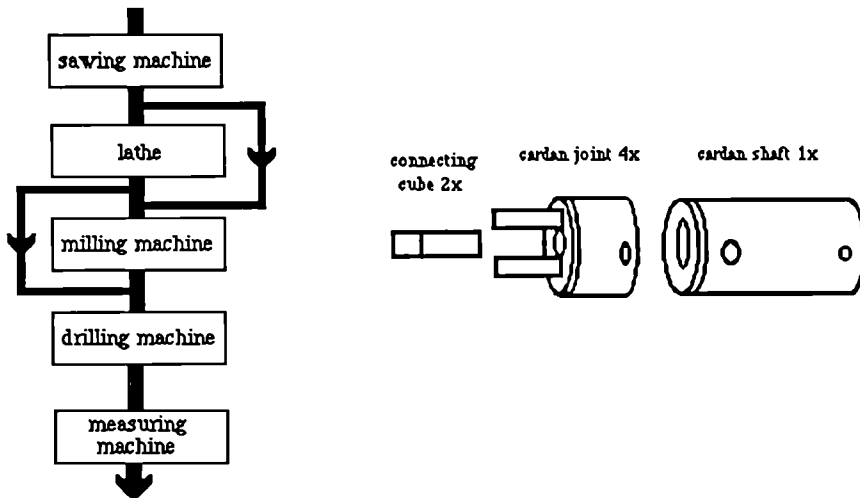


Fig. 10. Work flow through the FMS (left) and finished product (cardan drive)

The order file chosen is shown in table 1. The batch sizes are fictitious, but component quantities are in accordance with the number of assembled parts.

Order No	Name	Lot Size	Processing Time (sec)
1	cardan joint	160	26 040
2	cardan joint	160	29 840
3	cardan joint	160	48 440
4	cardan shaft	10	2 660
5	cardan shaft	20	4 600
6	cardan shaft	10	2 880
7	cardan shaft	10	3 520
8	cardan shaft	20	6 520
9	cardan shaft	10	3 840
10	cardan shaft	10	4 610
11	cardan shaft	20	7 550
12	cardan shaft	10	5 090
13	connecting cube	80	11 260
14	connecting cube	80	17 260
15	connecting cube	80	25 020

Table 1. Example Orderlist

6.2 Description of Strategies and Decision Criteria

Scheduling is done using global scheduling strategies. Scheduling can be done in two ways:

- if the buffers of the FMS are said to be of *First In First Out* type then scheduling only influences the order of the jobs that have not yet entered the FMS. Even then, the order of jobs leaving the FMS may be different from the order at the entrance, as different jobs may have different routes through the cell.
- if the buffers are of *random-access* type, then scheduling may influence the order of the jobs as long as they stay in a buffer

The implemented scheduling strategies can be subdivided in three categories:

- strategies that sort jobs according to some property, like the *Shortest Remaining Processing Time* (SRPT) rule
- strategies that only select jobs, like 'all jobs that have to pass through the milling-machine'
- combined strategies, like 'sort all jobs according to the SRPT rule in combination with the *Most Remaining Operations* (MRO) rule and raise the priority of all jobs that have to pass the milling machine'. The intention is to get this machine loaded as fast and as fully as possible.

Combined strategies are exhaustively used. For a list of basic strategies, see appendix 2.

The criteria by which a strategy is chosen are:

- optimization criteria like *minimal cost, maximal average workload, minimum flow time, etc.*
- error-criteria that occur when a machine breaks down or when a work piece is damaged and that try to load the remaining FMS as well as possible
- time-criteria mainly considering the state of the work period, such as *start-up, stable run and shut-down, second shift*

Error messages incorporated in the FMS-simulator are (the data sent when the error status occurs in parenthesis):

- machine breakdown (machine, expected time). If no breakdown cause is reported yet, an expected time is asked from the operator. If this time is below a preset threshold, no action is taken.
- machine slowdown (machine). Because of a fault such as tool wear tool wear, or because of some permanent deviation from normal operation.
- job too late (job number, machine, time). If the time is below a predetermined fraction of the processing time of the job, no measure is being taken.
- work piece breakage (job number, machine). No new order is entered in the system because of lack of storage. Reordering is not possible.
- product defective (job number). A new order is entered in the system, if reworking is possible.

Error messages reported by the simulator while checking the schedule are:

- queue overflow. There appears a bottleneck, so measures should be taken to relieve this machine.
- due-dates not matched. A different strategy is chosen, though this may be obsolete because it can happen that no schedule will match all due-dates.

State of the working period:

- start of working period: the FMS is (almost) empty and a strategy is used to fill it as fast as possible; with the presimulation of ASIM one can simulate the entrance of jobs in an already loaded

FMS, thus eliminating the *begin of workperiod* state to obtain typical steady state averages.

- stable run: the FMS is loaded close to the average loading percentage. Probably a different strategy is necessary now.
- termination of working period: this status is entered long before the end of the period. This state is especially important when it appears that not all work will be ready at the end.

6.3 Description of the knowledge base

The goal of the knowledge base is to maintain an optimal execution in, and loading of, the FMS. At the same time handle the operator's interaction.

At the beginning of a consultation, the expert system waits for an event from ASIM. If an event is transmitted, the alarm flags are investigated, and if no alarms occurred, no measure need be taken, and the expert system resumes the current consultation. However, if an alarm flag has been set, the expert system uses those rules in the knowledge base devoted to the selection of a set of candidate strategies. With this combination a forecast is run. If the forecast is successful, in the sense that during the forecast no alarms conditions have occurred, the orders in the cell are reordered according to the selected combination. If during the forecast, an alarm condition appears, a second set of strategies is selected and a second forecast is done. The result is treated in a similar manner as that of the first one. So a third try may be necessary. The result of the third forecast is treated differently to prevent a never ending sequence. If the third forecast does not succeed, then the orders in the cell are still rescheduled, but with the first set of strategies. The assumption behind this decision is that the first guess is likely the best under the current conditions.

The operator may interrupt the expert system at any time to get some information, to do a forecast or even a real scheduling by pressing a key on the keyboard. The expert system responds by prompting to the operator, who now is able to do several things. By using the standard Delfi-2 commands *how*, *why*, *rule* and *facts* he may ask information about the current consultation. Second, he may also ask for a forecast by responding to the prompt with *forecast*. Third, the operator can do a real scheduling by answering with *give_strategies*. In a real scheduling, the expert system asks for the preferred strategies and performs a regular scheduling. It does not prompt again. In a forecast the expert system also asks for the preferred strategies. It then performs a forecast but on completion it does prompt. The operator may again ask for more information, for a forecast or a real scheduling, or he may decide to do nothing. He finishes the session using the command *go*.

Example of rules taken from the knowledge base

Now we will present some rules from the knowledge base. Rules are presented here also in the form they are entered by the system configurator or by the cell operator.

IF

(the sawing machine is seriously down **OR** the sawing machine has a queue overflow **OR**
 the lathe is seriously down **OR** the milling machine is seriously down **OR**
 the drilling machine is seriously down) **AND**
 the lathe does not have a queue overflow **AND**
 the milling machine does not have a queue overflow **AND**
 the drilling machine does not have a queue overflow

THEN

the first strategy should be SNO

END

This rule belongs to a group of rules that determine the first set of strategies. It fires when the state of the FMS matches with the premise of the rule. This rule is applicable in many different situations, all having some common characteristics, so one can say that this rule treats a class of FMS-states and it is created from several rules of thumb used by operators in case of calamities.

The next rule decides whether a particular machine, the drilling machine, is seriously out of order. It is assumed that a machine shows a serious defect when its state is `out_of_order` and this will continue for a period longer than an acceptable limit. If any of the machines is defective, measures should be taken to adapt the FMS.

```

IF
  the state of the drilling machine is out_of_order AND
  the expected time this state will be as it is is greater than the acceptable limit
THEN
  the drilling machine is seriously down
END

```

This rule in its pure form states:

```

IF
  same drilling status out_of_order ;
  greaterthan drilling expected_time limit;
THEN
  conclude drilling down yes ;
  1.00
FI

```

7. CONCLUSIONS

We have demonstrated that the objectives of Work Package 2 of Esprit Project 809 are met. With the aid of the expert system Delfi-2, better production plans can be generated, especially when an FMS has not yet reached its steady state or lost this state. In those situations it is necessary first to explore several strategies. This is possible using the scheduler/simulator ASIM. Second, it appears that many deviations call for identical strategies. As soon as one of these situations is recognized, even an operator without programming experience can, if permitted, adapt the system by modifying or extending the rule base. This strategy is automatically used when this situation occurs again.

Dynamic scheduling with the aid of a real time expert system turns out to be feasible, although in order to gain more benefit from this approach, more constraints need to be incorporated in the simulation than is currently done, as it is expected that the limited availability of expensive equipment such as tools and fixtures that need to be shared between different machines is a serious source of inefficient utilization of an FMS today.

The effect of global scheduling strategies can be increased considerably when an individual strategy (or set of strategies) can be assigned to each machine in the FMS. The reason is that when a machine breaks down the FMS is often separated into several parts each with its own characteristics. Future research will concentrate on this subject.

Our system is clearly easier to adapt to different FMS than conventional systems. The simulator can be easily fed with another model. It only requires the simple operation of feeding the simulation with a new configuration. The knowledge base is, of course, dedicated to a particular FMS, but constructing or tuning a knowledge base is easier than altering the source code of an existing program.

Conventional scheduling strategies and simulation aids do not provide an optimal load of the cell in all cases. For that purpose more human control is needed. Initial loading of an FMS and rearranging of orders after an unexpected event are in particular need for new features. It appears that the possibilities for the operator to intervene in the process are not sufficient. He should be able to overrule the system and should have the opportunity to get more insight in the manufacturing process. During operation of the FMS, the operator know why a particular job is being dispatched instead of another, or what happens if a certain machine drops out. He may even want to assign a certain job to a specific machine at a given interval. A major concern in the near future will be the design of a powerful operator interface.

Some extensions of Delfi-2 had to be made. These extensions are made in the standard version of Delfi-2 and will be applied in other areas outside the scope of Esprit Project 809.

ACKNOWLEDGEMENTS

The authors wish to thank the team members D. Riewe and H. de Swaan Arons for their contribution and also the other consortium partners from Dextralog, ICL, and the University of Twente for valuable discussions.

8. REFERENCES

- [1] Blackstone J.H., Phillips D.T., Hogg G.L., A state of the art survey of dispatching rules for manufacturing job-shop operations, *Int. J. Prod. Res.* **20** (1982) 27..45
- [2] Ben-Arieh D., Knowledge based control system for automated production and assembly, in: Kusiak A., Modelling and design of flexible manufacturing systems, Elsevier 1986
- [3] Bruno G., Elia A., Laface P., A rule-based system to schedule production, *Computer* **19** (1986) 32..39
- [4] Fox M.S., Allen B.P., Smith S.F., Strohm G.A., ISIS: a constraint-directed reasoning approach to job shop scheduling, CMU-RI-TR-83-3, Pittsburg 1983
- [5] Gonzalez T., Sahni S., Flow-shop and job-shop scheduling: complexity and approximation, *Operations Research* **26** (1978) 36..52
- [6] Le Pape C., SOJA: a daily workshop scheduling system, in: Merry M. (ed), *Expert Systems 85*, Cambridge University Press 1985
- [7] Muth J.F. & Thompson G.L., *Industrial scheduling*, Prentice Hall 1973
- [8] O'Keefe R., Simulation and expert systems - a taxonomy and some examples, *Simulation* **46** (1986) 10..16
- [9] Shanker K., Tzen Y.-J. J., Loading and dispatching in a random FMS, *Int. J. Prod. Res.* **23**(1985) 579..595
- [10] Stecke K.E. & Solberg J.J., Loading and control policies for flexible manufacturing systems, *Int. J. Prod. Res.* **19** (1981) 481..490
- [11] de Swaan Arons H., Expert systems in the simulation domain, in: *Mathematics and Computers in Simulation XXV* (1983) 10..16

APPENDIX 1: Data Communication

Six typical data sequences appear in the communication between ASIM and Delfi-2. See table A1.

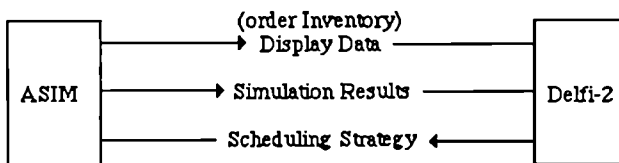
Communication	Source	Destination
Intermediate Simulation Results	ASIM-SU	Delfi-2
Forecast Results	ASIM-SU	Delfi-2
Final Results	ASIM-SU	Graphical Interface
Display Data	ASIM-D	Graphical Interface
Order Inventory	ASIM-D	Graphical Interface
Scheduling Strategy	Delfi-2	ASIM-SC
Trial Strategy	Delfi-2	ASIM-SC

Table A1. Communication Sequences

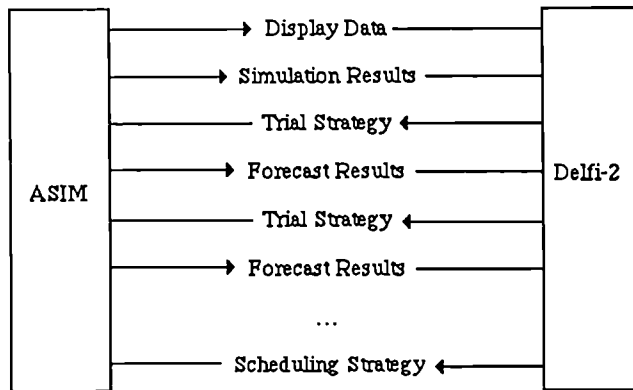
1. Display Sequence



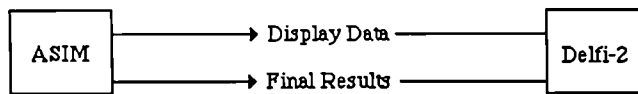
2. Consultation Sequence



3. Forecast Sequence



4. Final Sequence



During the consultation sequence at the end of the pre-simulation, the order inventory is sent instead of the display data. In case of alarm conditions, the transfer of display data preceding the simulation results is omitted.

APPENDIX 2: Strategies used in the Scheduler

EDD	Earliest Due Date
SPT	Shortest Processing Time
LPT	Longest Processing Time
SRPT	Shortest Remaining Processing Time
LRPT	Longest Remaining Processing Time
SNO	Shortest Next Operation
LNO	Longest Next Operation
FORP	Fewest Operations Remaining (for each) Part
MORP	Most Operations Remaining (for each) Part
MST	Minimal Slack Time
MSPO	Minimal Slack Per Operation
LATHE	enhance priority for jobs using the Lathe
MILL	enhance priority for jobs using the Milling-machine
DRILL	enhance priority for jobs using the Drilling-machine

Project No. 384

INTELLIGENT MODELS FOR ASSEMBLY DESIGN, PLANNING AND CONTROL

Dr. Stephen P. Sanoff, Dr. Barry Dwolatzky

GEC Research Ltd., Chelmsford, Essex, CM2 8HN, United Kingdom

ESPRIT project 384 - "Integrated Information Processing for Design, Planning and Control of Assembly" aims to demonstrate the principle of an integrated process for the development and manufacture of small batch assemblies.

A Target System has been defined to show how the functionality of a prototype system will be achieved. Central to this is the idea of having three models: an Equipment Model to describe the hardware available in the plant, a Process Model to describe what assembly processes may be performed on the equipment and Product Models to store descriptions of past products plus details of developing assembly designs. Using this approach, the system will be able to support a product developer in both generative and variant design and planning.

The paper concentrates on the Equipment Model and its first implementation in KEE (Knowledge Engineering Environment). The model seeks to provide all the information that may be needed for product development, for scheduling and for assembly station control. Clearly, each module in the target system has its own perspective on the data in the model and must therefore be responsible for its interrogation. Conversely, there are some aspects of the model that are general and which require computation or the application of heuristics for their use. In other words, a canonical knowledge representation is required such that there is no redundant information, but in which the model itself is intelligent enough to infer additional information that may be requested.

INTRODUCTION

In the design office and on the factory floor computers play an ever increasing role. A vast panoply of software tools - CAD and CAM packages, workshop schedulers, assembly planners, simulation programs - is becoming invaluable to engineers and production managers at all stages of product development and manufacture. CIM's objective of developing a single synthesis of these separate tools can only be achieved by the careful and correct integration of information flow within and between these functional aspects.

ESPRIT project 384 - "Integrated Information Processing for Design, Planning and Control of Assembly" - is investigating the information aspects of CIM within the context of flexible assembly. It is a 5 year collaborative project begun in April 1985. The partners in the consortium, lead by GEC Research (UK), are AEG (Germany), Telemecanique (France), MI-TNO (Netherlands) and IPK-Berlin (Germany). The project focusses on a conceptual Target System which provides a user with a knowledge-based environment (KBE) containing expert systems, databases and other tools. The system supports both generative and variant design of electro-mechanical products, and the assembly of small batches of these products within flexible robotic cells. The aim of ESPRIT 384 is not the detailed development and implementation of this target system, but rather the study of what information is needed, why it is needed, where it is stored and how it would flow within the system. To demonstrate concepts and to explore these information aspects some prototype elements of the target system are being implemented. Others (e.g. a CAD package) are being imported into the project from external sources.

Within the proposed system there are a number of 'models'. These are knowledge bases which are the repository of all information needed by, and generated within, the KBE. This paper concentrates on one of these models; it describes its role within the system and an initial implementation of it in KEE (Knowledge Engineering Environment). The model described is the Equipment Model which contains all of the required information concerning the factory, its assembly cells and manipulators, tools, etc., which are available.

The paper begins by describing the target system, outlining the information flow within the KBE. A knowledge elicitation exercise is then described which allowed the information content of the Equipment Model to be defined in some detail. An implementation philosophy for this model is then outlined based on a core of declarative knowledge and low-level inferencing. An object oriented structure for this declarative knowledge is specified. After describing an initial implementation of the model written in KEE, the paper discusses a proposed methodology by which it will be evaluated and refined.

THE TARGET SYSTEM

Figure 1 represents a schematic view of the target system. The KBE consists of six principal 'domain experts' which can be consulted by the user as the development and assembly of products proceeds. Each expert module within the KBE applies its expertise to knowledge which is stored in three models, namely the Equipment Model, the Process Model and Product Model(s). These are Knowledge Bases which contain, respectively:

- (i) details about the factory and its constituent hardware;
- (ii) the assembly processes which can be performed in the factory;
- (iii) descriptions of products which have been designed and can be assembled.

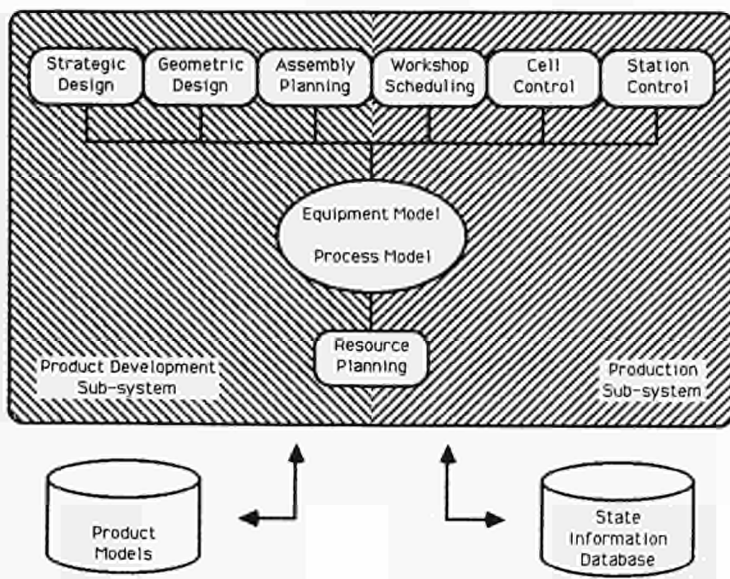


Figure 1: Target System

In the schematic the Process and Equipment Models are shown as being inside the KBE, whereas the Product Models are not. This was done to distinguish between 'read-only' information (i.e. information which is used only as a knowledge source within the KBE) and 'read/write' information which is both used and changed by the user with the help of the KBE. Ignoring for a moment the action of the Resource Planner, information concerning available processes and equipment is considered to be 'read-only', whereas the Product Models and the State Information Database (SID) are 'read/write'. The exception to this is when the system is consulted by the user in modifying the factory layout, in introducing new equipment, or in extending or changing the set of available processes. To assist in these activities the Resource Planner is used. In the schematic this expert module is depicted separately from the others as it is the only one which changes the information content, i.e. 'writes' into, the Equipment and Process Models. The other modules will only extract information from these models.

The SID is used and updated by the system in real-time as production in the factory proceeds. It contains, for example, information about the current position of components, pallets and grippers.

INFORMATION CONTENT OF THE KBE

Information evolves within the KBE as follows. When the target system is first commissioned within a factory, the KBE contains only the general expertise represented by each module. The entries in the Models are initially uninstantiated. The Planning Expert, for example, 'knows' everything it needs to know about general planning techniques, but has no information about the factory and what it can produce. The commissioning engineer consults the Resource Planner which assists in the selection and layout of equipment, and in defining the set of assembly processes available in the factory. We would not start with a completely 'clean slate', since the Resource Planner would, for example, have access to external libraries of equipment specifications. The Process and Equipment Models are thus instantiated with the required information. The product developer then begins designing new products. In this way Product Models are created.

Once product designs have been developed, production can begin in the factory. Based on the order-book the workshop scheduler creates a workplan. As components and sub-assemblies are brought into cells and moved about, the SID has its contents changed in real-time as the system keeps track of the production process. While production continues on the factory floor the system simultaneously allows the product developer to design new products, and variants of existing ones, which are added to the Product Models.

MODEL CONTENTS

We have said that the models within the KBE must contain all of the information required by the expert modules. A difficult question that has to be answered is: How do we determine which information must be stored? We wish neither to omit information which we need, nor to store redundant information. As a first step in the development of an Equipment Model, a knowledge engineering exercise was carried out. A number of brain-storming sessions were held in which a group of domain experts were asked to list everything they thought they might need to know about equipment in a factory. The exercise was carried out as part of the ALVEY "Design to Product" Large Scale Demonstrator Project. The result was a long, but unstructured, list (Reference 1) of information needed from an Equipment Model.

IMPLEMENTATION PHILOSOPHY FOR THE EQUIPMENT MODEL

A simplistic approach to model implementation would be to store this list within the KBE and to provide a mechanism for extracting information from it. If this were done two problems might arise:-

- (1) The implementation would be extremely large, resulting in a model which is expensive (in terms of computer resources) to store, and difficult to search efficiently.

- (ii) If, at some future time, an additional expert module is added to the target system, or equipment is added to or removed from the plant, *ad hoc* modifications to the model implementation would be required. The inclusion of a financial planning module is a possible example of such a change.

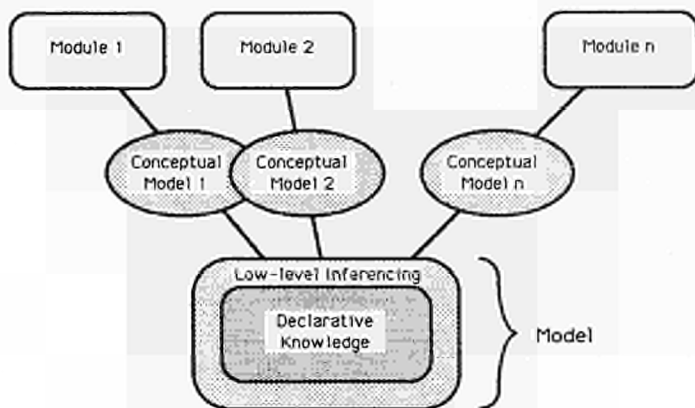


Figure 2: Distinct Conceptual Views of Model Implementation

Careful analysis of the list of required information led to an important insight into the role of the Equipment Model. Each domain expert has a different view of the factory and its equipment. To the designer of detailed component geometry, for example, a gripper has size and shape, whereas the cell controller sees it only as a resource with a given set-up time. In Figure 2 this representation of the model is depicted. The figure shows a number of expert modules, each having its own conceptual model. These models may intersect, implying that in some respects modules may have identical information requirements.

The conceptual models are not implemented as a body of facts. Each module has a set of rules whereby it can infer information about equipment using a core of declarative knowledge. This core is the minimum set of information, stored as a collection of facts, which will cater for the needs of all of the modules. Apart from its declarative knowledge the model also has some intelligence which allows it to perform low-level inferencing and computations. An example of this might be an ability to calculate the precise dimensions of a gripper finger as a function of temperature.

Implemented in this way the model is a 'canonical representation', in which the system contains no redundant knowledge, but allows the model itself to have sufficient intelligence to infer additional information that may be required. Development of the conceptual models is the responsibility of the designers of the expert modules. It is up to them to provide mechanisms which will allow them to answer their own questions about equipment.

Implementing the Equipment Model in the form depicted in Figure 2 requires a decision concerning the minimum set of declarative knowledge to be contained within the canonical representation. Using the unstructured list of requirements produced in the knowledge elicitation exercise a number of object-centred checklists were produced.

STRUCTURING DECLARATIVE KNOWLEDGE WITHIN THE EQUIPMENT MODEL

An object-oriented structure for the Equipment Model was decided upon. There are three classes of object:

- system level;
- machine level;
- component level.

Component level objects are usually made of a single material and have no actuation, for example simple fixtures and gripper fingers. Machine level objects have a number of different parts which can be made to move in a controlled way. Robots and grippers are obvious examples of these objects. Systems are both machines and components grouped together into assembly stations, cells and factories.

A detailed checklist was created for each of the object classes. Items in these checklists are structured by specifying related information under a single heading. The following is a small extract from the machine level checklist showing the level of detail required, and the internal structure of the list:

ENVIRONMENTAL CONSTRAINTS

Operating Temperature Range
 Operating Humidity Range
 Cleanliness
 Electrical Interference
 Lighting
 Safety

TIME

Set-Up
 Delivery
 Warm-Up
 Tool Changing
 Operation
 Maintenance
 Shutdown

The checklists, based as they are on the results of knowledge elicitation from a group of domain experts, should allow the Resource Planner to create the complete declarative knowledge base at the core of the Equipment Model. Not all of the items on the lists need be entered and stored in the model. It is possible to infer certain information given other information. If we know, for example, that a pallet has certain dimensions and is made of aluminium we can determine its mass, moment of inertia, etc. Additional low-level inferencing can then be added to the model implementation in an iterative process. As the expert modules develop their own conceptual views of the Equipment Model by devising their own interrogation rules, those which turn out to be widely applicable may be added to the Equipment Model itself.

AN INITIAL IMPLEMENTATION OF THE EQUIPMENT MODEL IN KEE

A first prototype implementation of the Equipment Model is being produced using KEE on a Symbolics Lisp Machine, as indeed are all the other models and modules of the target system.

KEE is a knowledge engineering environment which provides a number of AI tools, including: frame-based representation with multiple inheritance, demons, a rule system which supports goal and data driven reasoning, truth maintenance, message passing for object-oriented programming, a query and assertion language, a set of comprehensive graphics tools and access to the full power of the underlying Lisp language.

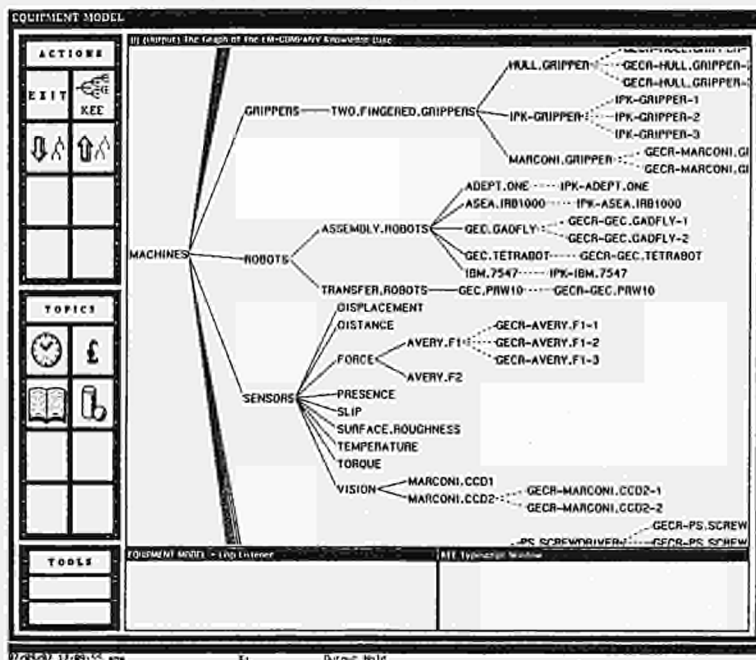


Figure 3: Experimental User Interface Showing Segment of Equipment Model

Figure 3 shows part of the Equipment Model hierarchy. KEE uses continuous lines to denote class relationships and broken lines to indicate 'instance-of' relations. For example, 'GEC. PRW10's are robots in the class 'transfer robots' and therefore inherit all the general properties of transfer robots. Conversely, the 'GECR-GEC. PRW10' robot is a particular, physical, machine which has a unique identity and location.

Most of the time, the model will be interrogated directly by the modules in response to the user's high-level assembly-related questions, without his even being aware of it. However, there will also be occasions when a user will want to inspect the model directly. In this mode of access KEE needs to behave like a sophisticated relational database, and the interface must shield the user from the complexities of KEE itself. The graphical user interface seen in Figure 3, is still experimental. It is aimed at satisfying the above requirements, supporting three broad types of interaction between the user and the model. The user may trigger an action, such as changing the hierarchy level being examined (e.g. cell, station or device); he may specify what topics are currently of interest, e.g. cost and time, or he may invoke special tools such as notepads, technical spreadsheets etc.

So far we have spoken of the Equipment Model as if it were a single entity but in fact it is convenient to implement it in two parts: a technological knowledge base (KB) and a company specific knowledge base. The Technological KB describes aspects of equipment which are always true. As an example, Figure 4 shows a small part of the classification of sensors (Reference 2) which is true for any plant. The Resource Planner implementing an Equipment Model utilises this, but is spared the effort of creating it. On the other hand, factory specific hardware is described in the company KB which contains all the instances of robots, grippers, etc., which describe the actual factory in which the target system is employed. It should be noted that the Technological KB constrains the Resource Planner to use the knowledge representation method employed and expected by the modules.

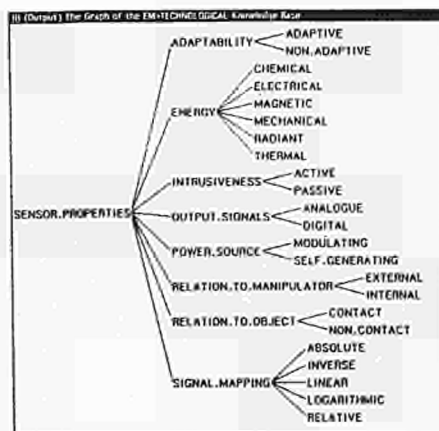


Figure 4: Example of Technological KB for Sensors

KEE supports this approach by allowing information to be inherited across KB boundaries. Thus, we can say that force sensor Avery. F1 (Company KB, Figure 3) is a NON.ADAPTIVE, MECHANICAL, ACTIVE, DIGITAL, MODULATING, INTERNAL, CONTACT, ABSOLUTE sensor (Technological KB, Figure 4). All the implications of this classification are then immediately available for reasoning about the sensor in question.

EVALUATION OF THE IMPLEMENTATION

As was pointed out in the introduction to this paper, the primary objective of the project is the study of information flow within an assembly-oriented CIM system. Our study of the Equipment Model and its implementation in KEE is aimed at furthering an understanding of which information is required within the target system, and how it might be efficiently used and structured. The fact that this target system exists only as a concept makes this study difficult. In other words, the only way of evaluating the content and structure of the Equipment Model is within a framework that does not yet exist.

A future development of the work described in this paper will take the form of a simulation game. Human domain experts will play the roles of the KBE modules and will have available to them software implementations (in KEE) of the Equipment and other models. A 'user' will be set design and production exercises which he will solve by consulting the simulated Target System. Each expert will have two note-pads, one for personal use and the other for passing messages to the models and other experts. By studying the information written on these pads, the performance of the system will be monitored and analysed. The first set of note-pads will display the details of the information required internally by each module, and the second set of pads will demonstrate communication between experts and interaction with the models. Thus the simulations will provide a methodology for studying information flow and for evaluating and refining the models.

CONCLUSIONS

The paper has described the utilisation of intelligent models in the factory of the future, with particular reference to ESPRIT project 384's work in the field of flexible automated assembly. With the aid of research programmes like ESPRIT, advances are being made in many areas intended to support the product developer and the production controller, resulting in new tools and methodologies. It is clear that knowledge based techniques are going to play an ever increasing role in these tools. With their ability to 'reason' about information, knowledge based techniques enhance and replace traditional numerical ones. Intelligent storage and retrieval of information is a major contributor to this enhanced software performance which the CIM community expects and deserves.

ACKNOWLEDGEMENT

We acknowledge the assistance received from the ALVEY 'Design to Product' Large Scale Demonstrator Project which carried out the knowledge engineering exercise described in this paper. The ALVEY Directorate is financing and coordinating collaborative research into information technology within the UK.

REFERENCES

1. Wright, R., "List of Data Generated by the Assembly Support Group", Alvey Design to Product Demonstrator, Ref: DTOP/PROJ/LCAV/14, March 1986, [Internal Project Report].
2. "Design Rules for Computer Integrated Manufacturing Systems", Version 1.2, Chapter 6, ESPRIT Project 34.

Development Towards An Application Generator For Production Activity Control

COSIMA Project Team

1 Introduction

This paper details the progress being made towards the development of an Application Generator (AG) for Production Activity Control (PAC). A previous paper [1] presented the COSIMA partners understanding of an AG for PAC. This paper presents the developments of the last year towards the overall goal of the project. The initial sections describe the main ideas of the AG and the functional blocks of Production Activity Control. The remaining sections describe software modules of the AG which have been implemented, namely:

- The Application Network
- The Manufacturing Profile
- The Test Bed Simulator
- The Rules Based Selection System

Each of these are distinct modules of the AG (the relationship between them is indicated in figure 1) and their integration will result an initial version of the Application Generator.

2 The Application Generator

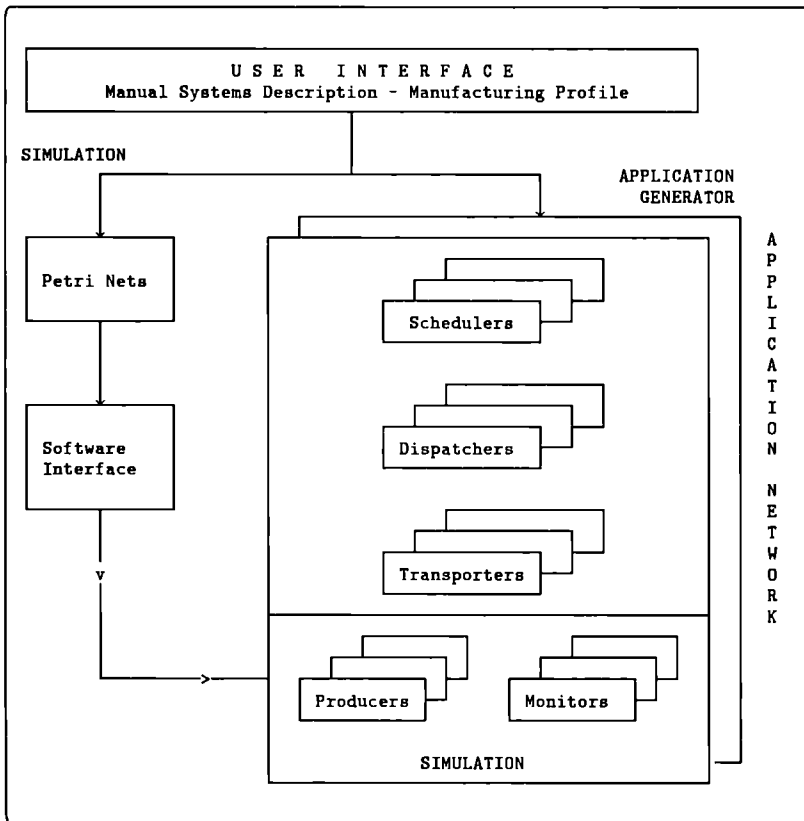


Figure 1: A view of an AG for PAC

Figure 1 documents the understanding of the major software deliverables of the Application Generator for Production Activity Control (PAC). This understanding is founded on the following basic ideas:

- Production Activity Control is composed of a number of basic building blocks namely:

- Schedule
 - Dispatch
 - Move
 - Produce
 - Monitor
- A simulation model of a manufacturing system can be developed from a Petri net model of that system. This is done through the development of software which produces an OPS5 [5] based simulator from a simple Petri net model.
 - The various models outlined above need to communicate through a communication system. This, in simple terms, is the role of the Application Network (AN).
 - A User Interface will *front end* the Application Generator. The User Interface will give access to a number of facilities, one of which will be to allow the user to present the *manufacturing profile* of the proposed PAC system to the AG.
 - The User Interface will also give access to a rules based system. These rules take as data the *manufacturing profile* provided by the user, as well as the answers to exploratory questions, and select the appropriate Scheduler, Dispatcher and Mover for the PAC system. This rules based system will have access to the simulator to help generate and test the proposed solution.

The AG will be a tool aimed at a manufacturing engineer who has intimate knowledge and experience in the factory for which the new PAC system is being designed. The AG will have a large suite of rules giving it knowledge about the building blocks (the software modules) for which it will choose the appropriate components for a PAC system. The AG has a very complex task to perform. Generation of a PAC system involves the performance of many diverse functions, from user input, testing, and simulation, to PAC specification output. All of these features will be available on a single machine through a single user interface.

3 PAC Functionality

Production Activity Control encompasses the control of the real-time operations on the manufacturing shop floor. Good PAC practice is a basic requirement for

manufacturing success and therefore, much attention has been centered on the area of PAC.

Any manufacturing organisation operates by utilising resources such as machines, materials and human operators. This activity is constrained by process data regarding the structure and form of both the product and the manufacturing process. Further to this is the order information from higher level business systems. The resultant outcome from the PAC activity is a set of instructions to the resources on the shop floor, and operational information to the various management functions.

One of the fundamental notions of COSIMA is the recognition of the five basic elements required to control production. The purpose of this separation, while not only being functional, was also to provide a common framework around which diverse manufacturing groups could develop and communicate a clear understanding of their manufacturing perspectives. Analysis of various manufacturing systems has helped us to develop a clearer understanding of these five building blocks. These functional subsystems within PAC are:-

1. Scheduler
2. Dispatcher
3. Mover Transport Control
4. Producer
5. Monitor

3.1 Scheduler

The Scheduler's purpose is the generation of a time sequenced operation plan for all of the resources to follow. This specifies the time scale for the transformation of raw materials into finished products by the assignment of resources to jobs at a particular time. Not only is this problem compounded by the choice of which strategy to apply, but also, by the availability of resources, cost of implementing decisions, due dates, customer priority, etc.. Different manufacturing systems require different schedulers incorporating scheduling policies suited to its own peculiarities, i.e. whether it

is a type of flow shop or batch shop etc.. Indeed, the manufacturing system's scheduler can well be composed of differing flavours of sub-schedulers for the cells and workstations. Each of these sub-schedulers is individually tailored to suit the objectives and characteristics of the sub-system.

3.2 Dispatcher

The scheduler plans for future resource allocations and the dispatcher attempts to accomplish this plan. There are two main entity types which the dispatcher controls, the movers and the producers. The dispatcher sends commands to these entities as regards what wip, raw material, tools and other equipment to move and what operations to perform next. It has to take into consideration the requirements for each of these operations as well as attempt to optimise the allocations of tasks. Different dispatchers can be chosen employing different dispatching techniques to carry out these tasks. One dispatching system could, for example, simply accept the ordering of jobs as fed by the scheduler, whereas another more sophisticated system, could use it as an input to further considerations as regards producer quality, interdependent setup costs at producers, alternative ratings etc.. A situation could even arise where the logic for job selection as applied in the scheduler is similar to that as applied within the dispatcher, both attempting to meet due dates and minimise inventory and flowtimes. The important distinction is that the scheduler generates a plan, whereas the dispatcher attempts to realise that plan in real-time.

3.3 Mover Transport Control

This functions task is to satisfy the requirements sent to it by the dispatcher. It manages all transport resources, i.e. Movers, (forklifts, agvs, trollies, operators), and will synchronise their operation to accomplish the movement of the item from its present location to the desired destination. It may be interactive, with human operators, or automatic, reacting and issuing commands on the receipt of feedback signals. Once the move is complete it signals back to the dispatcher and the location map of the material is updated.

3.4 Producer

The term producer can be applied to two types of entities, the operators who man the machines or the assembly stations and the software controllers on the process equipment itself. This is so because, PAC excludes the physical or process execution layer and communicates with it through the producers. The dispatcher communicates to the selected producer which operation it should execute on which job_wip item. The producer then downloads whatever instruction set is required and performs the correct setup for that operation. It collects performance data and communicates it to the Monitor function. When complete it instructs the dispatcher so that the WIP can be moved to its next operations.

3.5 Monitor

The Monitor collects data on materials, equipment utilisation and status and quality management and broadly speaking, acts as a real-time feedback mechanism to support decision making. From the materials point of view the floor inventory manager, within the Monitor, monitors the stock status on the shop floor. Quality levels are controlled and 'bad' material or processes signalled to the scheduler and the dispatcher. Equipment life, maintenance, performance and reliability are also managed. Outputs of the Monitor include notifications of system changes and decision support information to other levels, and reports to the user interface of the PAC system.

4 Application Network

The Application Network (AN) together with the Manufacturing Data Dictionary (MDD) provides a comfortable application environment for the building blocks of a PAC (Production Activity Control) system including all features that are needed to define the external interfaces of the building blocks in the PAC system and to enable communication between the building blocks via the so defined interfaces. The AN communication functionality includes intelligent data passing and data access mechanisms.

Each building block in such a system consists of one or more processes, which will be called Application Network User Processes (ANUP) in this document.

The communication mechanisms of the AN are based on the information stored in the Manufacturing Data Dictionary (MDD) and the AN database. The MDD contains the definitions of the information entities that are handled at the interfaces between the ANUP's and the AN. These information entities are called Data Transfer Units (DTU) in this document.

The AN database contains the information about the software configuration of an actual PAC system. The only external interface the ANUP has to handle is the AN interface. The ANUP doesn't know which other ANUP(s) can execute a certain operation or which ANUP owns a certain piece of data. This addressing information is stored in the AN database and in the MDD.

This allows the development of single ANUP's without needing to know the software configuration of the whole PAC system or the functionality of every other ANUP in the PAC system. Another advantage is the possibility of exchanging parts of the PAC system without affecting the implementation of the concerned ANUP's.

4.1 Functionality of the Application Network

The functionality described above, is fully implemented in the Application Network Base level 1.1 (AN BL1.1) version. The design of the next AN version will be based on the experiences encountered with the AN BL1.1 version. The functionality of this AN version is presently under study and the final results are not yet available. AN BL1.1 provides the following features:

- Definition of DTU's.
- In the current version, AN and MDD provide a common definition interface to define the DTU's used in a PAC system and to define subscribers and owner(s) of a DTU in this PAC system.
- This interface is the tool to tailor the configuration of a special PAC system from the communication point of view.

- Connecting to/ Disconnecting from the Application Network

Before an ANUP can use the AN communication services, it must connect to the Application Network. When the call is confirmed, the ANUP-specific initialisation has finished, the ANUP can issue requests to the AN and can receive information from the AN.

The ANUP can disconnect from the AN in case of errors or when it has finished its communication activities.

- Information Multicasting

An ANUP can provide a DTU to the PAC system via the AN. The DTU is then forwarded to the ANUP's of the PAC system according to the information in the AN database. The sending ANUP gets information about any errors during the delivery of the DTU.

- Information Retrieval

An ANUP can request a DTU from the PAC system via the AN. The AN then determines the ANUP which owns this DTU. The information about the ownership is stored in the AN database. This ANUP is called by the AN and returns the requested DTU which is finally delivered to the requesting ANUP.

- End-to-end delivery guarantee

Because there is no End-to-end peer relationship between the ANUP's, the AN insures that data delivery is guaranteed, or, if any error occurs, informs all interested parties about non-delivery.

4.2 Actual state of the implementation

The described functionality is fully implemented in AN BL1.1. This AN version has been delivered to the partners within COSIMA. In the existing building blocks that have been implemented, the previously used communication mechanism (IS) has been replaced by the AN BL1.1 version. The substitution did not create any major problems.

5 A Manufacturing Profile Prototype

A prototype of the Manufacturing Profile (MP) for describing manufacturing systems has been implemented in the database language VAX Rdb/VMS. The data are stored in a series of named fields. The fields are organised in groups called relations which represent the entities in the system.

The data contained in the database is divided into four sections. They are;

- **Planned Orders:** This is information on the products required, their due-dates and the quantity required.
- **Capacity information:** Consists of information on available resources on the shop-floor, including machines, operators and transportation devices.
- **Processing Data:** Data on the products to be manufactured, their routes, processing times etc. Where a main process is an assembly operation a field refers to the part or parts which are required for that assembly.
- **Information concerning Materials:** This specifically refers to processes which make assemblies to be used in the main processes.

Any number of different profiles can be stored using the same data structure. The system includes an editor for modifying existing profiles and specifying new ones. The data structures are constantly being expanded to allow for the inclusion of new characteristics or new descriptions. The present version is intended to act as a prototype on which more complete versions of the MP can be built.

6 Test Bed Simulator

The Test Bed Simulator is an initial attempt to define the features of part of the Application Generator. The part of the AG it is specifically concerned with is the *Simulator* module. As can be seen from figure 1, the Simulator (which is based on Petri net theory) takes its input from the manufacturing profile, and then links with the other PAC building blocks to allow the user generate a simulation of their

manufacturing process. The Test Bed Simulator consists of the following modules (the Manufacturing Profile module is based on the description given in the previous section):

- Manufacturing Profile
- Simulator
- Dispatcher
- Monitor
- Communications Network

6.1 Simulator

As the AG cannot be run with a live system, a Simulator is needed to provide the characteristics of the production system. The Simulator may then provide the AG with the ability to be applied to varying manufacturing situations.

The goal is to be able to produce a generic simulator which will integrate with the other modules of the AG. The role of the simulator is to simulate the flow of work through the shop floor. It is not a conventional simulator, because it does not have the control strategy and entity simulation in the same module. The control of the flow of work is provided by a separate module, the Dispatcher, and the Simulator simulates the actions of the producers and the Monitor. The Simulator itself is based on the theory of Petri nets.

6.2 Dispatcher

Dispatching is the final determination of job sequencing for a workcenter and it is responsible for coordinating the individual workcenter schedules, the workcenter itself and material movement control. The Dispatcher receives information from the Simulator and then acts accordingly. The types of messages the Dispatcher receives are the following:

- A job has arrived into the manufacturing system.
- A job has finished on a machine.
- A moving device has arrived at its location.
- The status of the moving devices.

The Dispatcher will then decide what has to be done next. It does this by referencing the route file. The Dispatcher, which is written in OPS5 will issue the following commands to the Simulator.

- Move a job from location i to destination j .
- Send a job out of the system.
- Dispatch a job from a moving device to a work station buffer.

6.3 Monitor

The Monitor makes a log of all of the events which occur during the simulation run, and then it collects statistics on machine utilisation and throughput time. It uses a database to store the relevant data.

6.4 Communications Network

The role of the communications network is to provide a transport mechanism for the data flow between the different modules. The goal of the Application Generator project is to produce an *Application Network* (see section 4) which will take care of all the data flow. The data flow is handled by a software package which uses mailboxes to send messages from one process to the other. The communications network which was used for the Test Bed Simulator has now been replaced by the Application Network.

7 A Scheduler Selector Prototype

The scheduler selector prototype is in the form of an OPS5 rules base, which extracts some of the information from the MP and advises on key scheduling areas, based on that information. The rules concentrate on both reducing the problem to more manageable proportions by identifying patterns within the manufacturing system, and on what are perceived to be the most critical resources in the system. The topics considered are;

- The process ratings for the different products are examined and compared to assign a degree of flow, on a scale of 1-10, to the system.
- Pre-scheduling Capacity check; Based on the planned orders a decision is taken on whether the system will have the required capacity to fill those orders. Demand is allowed to reach a predetermined percentage of the total available capacity. The percentage allowed is scaled according to how smooth the product flow is within the system i.e. the degree of flow assigned.
- One and Two machine problems; Standard solutions to one and two machine problems are recommended where the the manufacturing system description indicates one or two machines, or where problems are reduced to these proportions.
- Machine Groups; The user is informed as to how a group of machines may be considered together in order to reduce the complexity of the problem. Then the user is asked whether such groups exist within the system, and the problem is reduced appropriately.
- Bottlenecks; Advice is given on how bottlenecks should be handled to improve the flow of products through a system. Both single and twin bottleneck cases are considered.

Later versions of the scheduler selector will typically suggest a range of possible solutions to be run in the simulator, and based on the results from the simulator recommend a particular scheduler.

8 Conclusion

This paper has presented the developments of the COSIMA project since July of last year. The major achievements since then have been:

- The development of the Application Network which is now capable of sending messages from one PAC software module to another.
- The development and integration of the Simulator with the other modules of PAC to form the Test Bed Simulator. The Test Bed Simulator can take the representation of a manufacturing system through the Manufacturing Profile and develop a simulation model from this data. This simulation model simulates the flow of work through the manufacturing system, while the Dispatcher provides the control strategy to route work through the system. The Monitor building block collates information on system performance.
- The development of a Manufacturing Profile and a prototype Rules Base Selection System. This system analyses the information stored in the Profile database and suggests a scheduling strategy incorporating the important parameters of the system. It is based on the OPT paradigm checking for bottlenecks which may occur in the system and producing suitable solutions.
- The integration of the Application Network with the Test Bed Simulator. This has been achieved so that the four separate software building blocks of the Test Bed Simulator can now communicate using the Application Network.

These are important modules in the development of an Application Generator for Production Activity Control.

References

- [1] **Mc Cahill, John.**, *Towards An Application Generator For Production Activity Control In A Computer Intergated Manufacturing Environmeny*. COSIMA, Esprit Project 477, June 1986.
- [2] **Peterson J.L.**, *Petri Net Theory and the Modeling of Systems*. Prentice Hall Inc., Englewood Cliffs, NJ 07632.

- [3] **Bruno G. and Marchetto G.**, *Rapid Prototyping of Control Systems using Petri Nets*. Dipartimento di Automatica e Informatica, Politechno di Torino, Italia.
- [4] **Harhen J. and Browne J.**, *Production Activity Control: A Key Node in CIM*. International Working Conference on Strategies for Design and Economic analysis of Computer Supported Production Management Systems, Vienna, Austria 28th-30th September 1983.
- [5] **Scown, S.J.**, *The Artificial Intelligence Experience: An Introduction*, Digital Equipment Corporation, U.S.A, 1985.
- [6] **Jarvinen, Ora and Konrad, Hermann.**, *AN BL1 Specification (Draft)*, Esprit Project 477, COSIMA, January 1987.

List of Acronyms

AG	Application Generator
AN	Application Network
ANUP	Application Network User Process
COSIMA	Control Systems for Integrated Manufacturing
DTU	Data Transfer Unit
MDD	Manufacturing Data Dictionary
MP	Manufacturing Profile
PAC	Production Activity Control
WIP	Work In Progress

Prepared By:

Comau spA,
Torino, Italy.

Computer Integrated Manufacturing Research Unit (CIMRU),
University College, Galway,
Ireland.

Digital Equipment Corporation B.V.,
Clonmel, Co. Tipperary,
Ireland.

Digital Equipment GmbH,
Munich,
Federal Republic of Germany.

RENAULT Automobiles,
Boulogne Billancourt Cedex,
France.

ARTIFICIAL INTELLIGENCE IN PRODUCTION SCHEDULING

M. CLARK
F. FARHOODI

LOGICA plc, 64 Newman Street, London W1A 4SE, UK

This paper describes some of the work performed within Project 418 [1] in the area of Machine Shop Scheduling tools which are a combination of algorithmic and artificial intelligence methods.

1. INTRODUCTION

1.1. Overview of Expert Systems

Until recently computers have been used in applications where human capabilities are limited - high speed calculations and storage and retrieval of large volumes of data. Such computer systems are based on algorithmic programs involving completely defined step-by-step procedures for solving problems. The emergence of Expert Systems and, more generally Knowledge Based Systems (KBS), during recent years has shown that it is now feasible to devise computer programs which are capable of using non-numerical information in a manner that emulates the way human experts employ knowledge and experience. These systems provide advice and explanations as well as results in numerical form. The necessary software and hardware to produce Knowledge Based Systems is becoming commercially available.

Knowledge Based Systems are still at a formative stage and many of their potential capabilities are often over-stated. Nevertheless, it is widely accepted that, at last, Artificial Intelligence (AI) has generated some novel concepts and techniques which will gradually be evaluated, refined and utilised by the computing industry at large. In particular, the advent of KBS techniques has motivated many companies to examine the ways in which they treat knowledge and expertise as a manageable resource separate from the people who presently possess the knowledge.

Although the capabilities of current systems are severely limited in many respects, small-scale applications of KBS techniques are already in operational use as Decision Support tools associated with the well defined task domains. The initiative for the development of many of these systems has come from the "user departments" of organisations. Such departments are usually interested in structuring the knowledge and problem-solving skills of their experts or specialists in a particular task, so that the knowledge can be made explicit and more accessible to less experienced staff, often for training purposes.

The main technical issues in the development of AI systems are as follows:

- identifying the scope of the knowledge used by experts in problem solving situations;
- establishing ways of efficiently eliciting the knowledge of a human expert in a given field;
- defining methods to fully and unambiguously represent and describe knowledge in all its forms - normally referred to a knowledge representation;
- implementing ways to efficiently process this knowledge to solve real life problem situations.

2. ARTIFICIAL INTELLIGENCE AND PRODUCTION SCHEDULING

2.1. Current Scheduling Systems Involving Artificial Intelligence

The principal work that has been done in the area of Artificial Intelligence as applied to shop scheduling is connected with the "Factory of the Future" project at Carnegie Mellon University in Pittsburg, USA. Two systems have been developed in association with this programme: ISIS, the Intelligent Scheduling and Information System; and OPIS, the Opportunistic Intelligent Scheduler.

ISIS^[2] was the first of two developments, and was started in 1980. It was aimed at applying knowledge based techniques to scheduling in the job shop environment. The scheduling approach taken is order based whereby the shop schedule is generated on an incremental basis dependent on order priority, but also constrained by capacity and resource limitations. Essentially the schedule is built up by repeatedly selecting and scheduling those outstanding orders which have the highest priority. This process is repeated within the constraints of the situation to generate the final schedule.

OPIS ^[3] was a development of the ISIS project, with particular emphasis on solving a weakness of the ISIS system in respect of conflict oriented scheduling situations. OPIS builds on the ISIS experience but is aimed at scheduling in situations where there are bottleneck machines. A dual approach is taken by OPIS: the first stage involves generating and fixing a schedule for the processes on the bottleneck machine(s); the second stage completes the scheduling by dealing with all the outstanding processes and machines.

An overview of work done at Carnegie Mellon University is presented in Reference ^[4] .

2.2. Role and Expected Benefits of Expert Systems in Scheduling

The job shop in the manufacturing industry has traditionally been a difficult environment in which to achieve high levels of productivity. The reasons for this have centred around the problems associated with production planning and control, an in particular with the difficulties of effective shop scheduling.

The construction and maintenance of good schedules has been a problem due to:

- the complexity of the manufacturing environment in both its operational and engineering aspects;
- the difficulty of resolving conflicting preferences;
- the difficulty in resolving multiple preferences;
- the problems in translating business goals into operational plans;
- the limited use of real time monitoring to progress the schedule;
- the problems of constraints which ideally should be treated as preferences;
- the difficulties of re-scheduling due to unforeseen disturbances (machine breakdowns, etc...).

In most practical situations - even those which employ computer based scheduling methods - the human production scheduler would need to take account of many of these factors to varying degrees. Indeed, traditional computer based scheduling methods only incorporate a small fraction of the scheduling knowledge associated with a manufacturing situation. Hence the schedule generated is a high level guideline for the shop supervisor or foreman who will then be responsible both for detailed scheduling, and also for maintaining the schedule during the period in question. This may be feasible to do in small shop, but in a large scale facility the complexity of this task becomes burdensome to undertake.

The objective of using AI techniques in scheduling is to incorporate more of the pragmatic scheduling knowledge into the computer based scheduling process. This will facilitate the generation of schedules which more fully reflect the goals of the business, as well as more appropriately understanding the operational considerations of the shop environment to which they are applied.

3. EXPERT SCHEDULE IMPROVEMENT SYSTEM

In order to test the validity of these concepts, Project 418 has developed a Scheduling System for evaluation and experimentation purposes. The main characteristics of this system are described below, along with the general methods used to derive and structure the relevant knowledge elements.

3.1. Architecture

3.1.1. Overall System - The A.I. Scheduler

The function of the Scheduler is to provide a decision support tool for the management of a job-shop environment. The system (Fig. 1) is required to perform the following operations:

- generate production schedules using an algorithmic (non-AI) approach;
- evaluate schedules generated by the algorithmic technique against expert knowledge of potential problems;
- proposed corrective actions to repair faulty or "non-optimal" schedules;
- interact with the user before desired corrective actions are carried out;

- allow the user to inspect and modify generated schedules manually (without using the schedule evaluator provided by the system);
- provide facilities to the user for:
 - . defining new shop configurations,
 - . inspecting or updating data relating to various shop objects/entities (machines, operators),
 - . inspecting generated schedules and information on resource utilisation,
 - . interfacing with graphical displays.

3.1.2. Inputs and Outputs

Inputs:

The inputs to the Scheduler fall into three categories:

- session-independent information, ie data on the inherent characteristics of objects/work-shop entities such as:
 - . operators (names, skills), operations (type, description);
- session-dependent. These are shop configurations information for a scheduling session, for example:
 - . product/order descriptions and order priorities.

Outputs:

- schedules for:
 - . machines (showing orders processed by each machine over a given period),
 - . operators (showing orders processed by each operator over a given period);
- information on characteristics of various shop object/entities.

3.1.3. Scope

The Scheduler will initially operate as a standalone system, with no facilities for interfacing with other computer systems.

The user will be able to interact with the Scheduler using a mouse, restricted natural language (for database operations only) and keyboard.

To summarise, the Scheduler will allow the user to perform the following operations:

- interact with the Scheduler graphics by using a mouse;
- interact with the Scheduler database system by using menus and restricted natural language;
- define new shop configurations and alter/add to the characteristics of database items;
- modify generated schedules without the aid of the system schedule evaluator;
- define/change constraints at run-time.

3.1.4. The Scheduling Process

At the highest level the user should be able to use the Scheduler for two purposes:

- interrogating/updating the permanent database (defining new operators, machines);
- generating production schedules.

The scheduling process would start by the user supplying a number of goals (manufacturing orders to be scheduled) and constraints (eg availability of machines). The system would then generate the first round of schedules for evaluation by the user or the system. The results of the scheduling process would be displayed to the user in a graphical format (under user command).

The users decide whether the system or themselves evaluate the schedules. If the user selects system evaluation, the schedule improvement sub-system would examine the schedule using its expert knowledge of potential problems and propose corrective actions (if any problems are detected).

The user would then decide if all or part of these actions are carried out. Once the corrective actions have been executed a new schedule is produced.

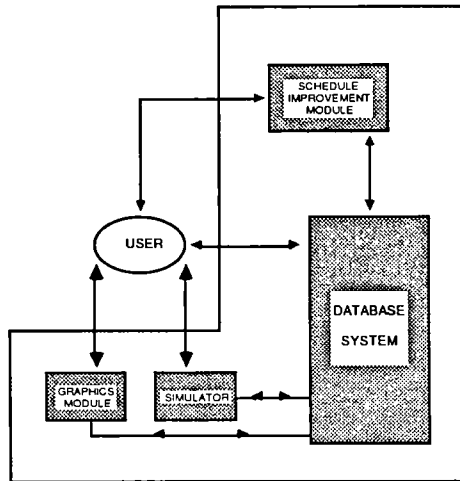


FIGURE1
Architecture of the Scheduling System

3.2. Structure and types of knowledge in schedule improvement

Generally speaking, schedule improvement knowledge falls into three categories:

- general (generally accepted methods/rules);
- domain specific (these modify/extend generally accepted methods in a particular domain);
- scheduling technique related.

Knowledge about the characteristics (weaknesses, strengths) of a scheduling technique can be either explicitly represented (for examples as rules to influence the choice of corrective actions) or implicitly be taken account of in designing the remedy rules (in the context of Esprit Project 418 the latter approach has been adopted).

Domain specific knowledge affects not only the composition of problem, cause, remedy categories and assignments of priorities, but also the method (steps) used for implementing the corrective actions, or in general the control of the reasoning process.

Classification of Knowledge

The classes of knowledge which have been identified as essential in the schedule improvement activity are described as follows:

• Objectives

Objectives relate to the goals of a manufacturing environment; these relate to different activities (resource planning, process planning etc...) and apply at different levels (factory, shop, etc...). At the highest level, goals such as profit maximisation apply, whereas at the factory and shop levels goals are more closely tied to the production processes.

Each decision making level receives a set of constraints and goals from one or more levels higher which then form the criteria against which the performance of this level is assessed.

• Objects

Knowledge about the types and characteristics of objects in the shop plays an important part in the schedule improvement process, especially in the selection of corrective actions and the relaxation of constraints.

* Problems and Causes

Problems are situations or conditions which impair the achievement of production objectives at the shop level. Some of these problems are caused by bad scheduling, whereas others can originate from activities such as process planning or resource planning.

Causes are basic problems which combine to create more complex problems. A superficial distinction is made here between "*high level*" and "*basic*" causes. High-level causes can be directly associated with problems; these are themselves brought about by one or more other causes. Basic causes are faulty conditions which the corrective actions are designed to remove. Basic causes can be related to a responsibility/authority level in the shop.

It is possible to analyse some problems at different levels of aggregation. For example, lateness can be examined with respect to a particular batch-order or all batch-orders. This distinction is important because the appropriate scope/severity for applying corrective actions in relation to a given problem can be greatly influenced by the level of aggregation at which that problem is being considered.

* Corrective Actions

Corrective actions are mainly constraint relaxation actions which are designed to remove "*basic*" causes of problems. It is necessary to associate with each action one or more levels of authority, indicating its severity; the implementation of each action would then need to be sanctioned by the appropriate level of authority.

It is important to be aware of potential interactions among corrective actions, namely "*side-effects*"; otherwise, the Scheduler may spend a great deal of time on correcting for side-effects of its own actions let alone removing the original problems.

For each action there is a minimum prerequisite amount of knowledge, eg applicability condition, problems the action can help cure, steps involved in implementing the action, etc...

* Evaluation Functions

These are tests which determine the following:

- whether a problem/cause has occurred (and to what extent);
- amount of improvement required, in relation to a detected problem;
- the degree of success achieved by corrective actions.

Evaluation functions need to be defined for both "*specific*" problems and "*average*"/overall problems. Evaluation functions can consist of simple arithmetical equations or heuristic rules.

* Constraints

Knowledge about constraints can be examined from different view points, these include:

- significance/degree of influence of each category:
 - . performance (eg due-dates, WIP, levelling, costs),
 - . physical (eg set-up times, tear-down times, processing times, machine physical constraints),
 - . causal (operation sequence/routing alternatives, machine alternatives, materials and personnel requirements, transport-time),
 - . availability (eg machine exceptions - maintenance);
- type of representation:
 - . factual statements,
 - . absolute - comparing a parameter against a fixed threshold,
 - . relative - comparing one parameter against another of same type,
 - . rules, these relate two or more parameters of different types (eg if component type X needs welding then use a type Y welding machine).

The initial limits/thresholds (absolute, relative or as defined by rules) for constrained parameters need to be obtained for each manufacturing environment; values can be defined for many of the parameters on an ad-hoc basis but, this is likely to lead to erroneous decisions. It is necessary, therefore, to acquire these threshold values from experts in the selected domain and/or optionally elicit "*configuration*" rules from experts for types of domains (eg machine job shops), in order to be able to compute these thresholds, as well as relative importance or priorities of each type of constraint, in actual instances of such domains.

Generally, the following factors influence the limits/thresholds for various constraints in machine job shops:

- level of automation,
- type of processing (component, assembly),
- stability in product range,
- order controlled or inventory controlled operations
- product characteristics,
- cost structure,
- quality,
- complexity of production.

* Problem-solving Strategy

The strategy used by an expert to assess a generated schedule against selected criteria, and the way he/she selects and applies corrective actions have a crucial impact on the manageability and efficiency of the schedule improvement process.

Knowledge of the "*correct*" strategy is important in a number of situations, for example:

- selecting the most important criteria, to evaluate first (the simplest way is to use a "*default*" ordering);
- choosing the problem level. Having selected a criteria such as "*meet-due-date*", deciding on which evaluation functions to apply, or alternatively which aspect of which problems to look for (in this example, we can look at "*average lateness*" first or check a particular batch-order for lateness;
- selecting the focus set. Having selected an aspect of the problem (level of abstraction or detail), we need to decide which sub-set of the affected items to examine, and in which order to go through this sub-set.

3.3. Framework and Results of Knowledge Elicitation

Knowledge used to construct paper and computational models of the "*expert*" decision making process was derived using a formal framework, as follows:

3.3.1. Objectives

The scope of this analysis is confined to shop-level scheduling. At this level, the following objectives have been identified:

- meet-due-date (C1),
- maximise-utilisation-of-equipment (C2),
- minimise-inventory-time (C3),
- minimise-throughput-time (C4),
- minimise-costs (C5).

3.3.2. Objects

The list of shop objects identified so far, together with the attributes/characteristics defined for each is in accordance with the generic data model derived as part of the OCS architecture.

3.3.3. Problems

The following high-level problems and their variations have been identified in relation to each of the above objectives:

- scheduled-finish-date-too-late (C1)
 - . all-batch-orders,
 - . specific-batch-order;
- scheduled-finish-date-too-early (C1)
 - . all-batch-orders,
 - . specific-batch-order;
- utilisation-of-machines-too-low (C2)
 - . all-machine-types,
 - . specific-machine-type,
 - . specific-machine;
- inventory-time-too-long (C3)
 - . all-batch-orders,
 - . specific-batch-order;

- throughput-time-too-long (C4)
 - . all-batch-orders,
 - . specific-batch-order;
- costs-too-high (C5).

Generally, it is considered that global problems (ie those affecting a number of shop items) should be examined before local ones. Figure 2 depicts the causation tree for the first of the problems listed above, including variations. The following types of information have been elicited with respect to each problem or cause:

- other problems it can cause;
- criteria it relates to;
- evaluation functions for determining existence and severity;
- selection function for identifying objects affected;
- possible causes;
- possible remedial actions;
- problems with which it is mutually exclusive;
- a measure of seriousness.

3.3.4. Corrective Actions

A number of types of corrective actions have been suggested by the domain experts. A full list of these actions together with cross references to the appropriate decision-level has been derived as a result; a subset of these actions is incorporated in the Scheduler.

The following types of corrective actions are typical of those identified at this stage:

- change batch-order priority;
- change shop capacity:
 - . available time,
 - . number of machines.

3.3.5. Evaluation Functions

Evaluation functions/rules (ranging from arithmetical formulae to more complex heuristics) for the majority of problems, causes and actions have been defined by the experts.

3.3.6. Constraints

In the context of schedule improvement, knowledge about constraints is used mainly to guide the following processes:

- evaluating problems:
 - . checking whether a problem has occurred, by examining values of relevant parameters against thresholds defined by experts,
 - . identifying the scale of a problem, by assessing the degree by which a given parameter exceeds specified thresholds;
- selecting corrective actions and the desired magnitudes;
- assessing the costs v benefits of proposed corrective actions.

The most significant constraints identified so far are those which relaxation can have a noticeable influence on the process of schedule generation.

3.3.7. Problem-solving Strategy

No expert knowledge has been elicited in the area of problem-solving strategy; however, default rankings for criteria (in terms of importance), and causes (in terms of likelihood of causing each problem) have been suggested by our experts.

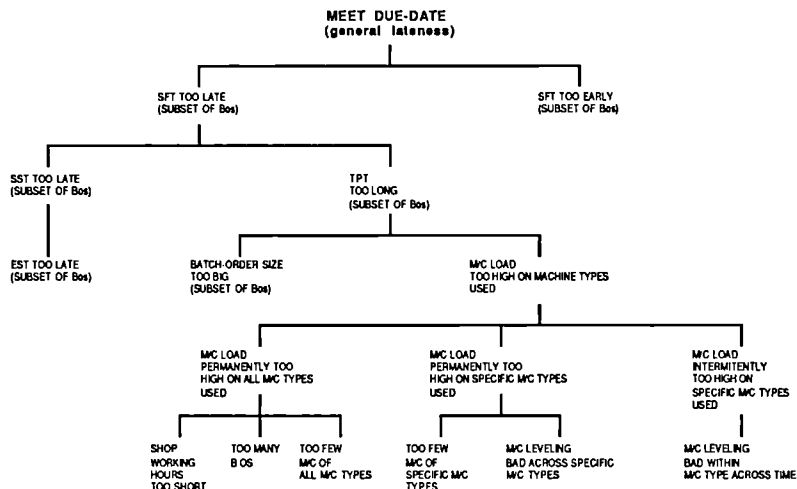


FIGURE 2

4. RESULTS AND CONCLUSIONS TO DATE

4.1. Results

The actual implementation of the Expert Scheduler has been performed using a standard IBM PC/AT; the software (see figure 1) uses PROLOG 2 for all system components except the actual algorithmic scheduler (*"The Schedule Simulator"*), which uses the procedural language "C". This configuration was used for a number of technical reasons, and is not one which would lend itself to a practical implementation in a manufacturing environment. Nevertheless, the main results emerging from this exercise would, we believe, be valid in any implementation context, namely :

- Although there have been a number of problems in constructing the prototype, we believe that, on the evidence to date, the main objectives have been met; an algorithmic schedule, which can subsequently be improved by recourse to knowledge based methods.
- The hybrid approach has resulted in some overlap and potential duplication of data at the boundary between the traditional manufacturing database and the knowledge base (in particular, where objects within the shop are concerned). Although to some extent this is a function of the implementation methods, it is nevertheless true that representation of factual information has to be handled carefully if inconsistencies are to be avoided.
- Some forms of knowledge in a CAM environment (objectives, evaluation functions and, to a lesser extent, problems) require a considerable amount of repetitive browsing of the system database, and subsequent procedural computation before artificial intelligence mechanisms can be deployed (eg "What is the value of work in process"). This is not strictly the domain of normal A.I. tools.

4.2. Conclusions

- Hybrid systems for machine-shop scheduling are feasible, and, properly engineered, can represent a generic solution to the scheduling problem which is open and flexible.
- Careful design of the interface between traditional system database and the domain-specific knowledge base is essential; we believe that the starting point should be a consistent data model of the target system which can be expressed in the requisite formalisms on each side of the hybrid system. Thus, for example, the conceptual data model might be translated into a semantic network for A.I. purposes.
- There are limitations to the amount and scope of high-level, dynamic "knowledge" which can be embedded in the expert components of a hybrid model. It seems likely that a "computational layer" will be required which will act as a server to the expert system, thereby representing formally the boundary between data and information.

REFERENCES

- [1] Esprit Project 418 : *Open CAM System*, Proposal to the Commission of the European Communities, 1984
- [2] Fox, M.S. and Smith, S.F. : *ISIS: A Knowledge Based System for Factory Scheduling*, in *Expert Systems* 1, pp 25-49, 1984.
- [3] Smith, S.F. and Ow, P.S. : *The Use of Multiple Problem Decompositions in Time-Constrained Planning Tasks*, in *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 1985.
- [4] Smith, S.F.; Fox, M.S. and Ow, P.S. : *Constructing and Maintaining Detailed Production Plans : Investigations into the Development of Knowledge Based Factory Scheduling Systems*, *AI Magazine*, pp 45-61, Fall 1986.

Project No. 932

ASPECTS OF KNOWLEDGE-BASED FACTORY SUPERVISION SYSTEMS

Wolfgang Meyer, Randolph Isenberg
Philips GmbH Forschungslaboratorium Hamburg,
Vogt-Kölln-Str. 30, D-2000 Hamburg 54, FRG.

The paper presents a unified methodology for analysis, design and implementation of AI based software modules, like expert systems for production planning, quality control and preventive maintenance, for the factory of the future, both large and small batch-oriented. An extension of the hierarchical NBS factory reference model towards a distributed intelligent controller model is proposed. Planning horizons and planning update periods determine the hierarchical levels whereas decision frames given from upper to lower levels are regarded as constraints determining a certain degree of self responsibility for each separate controller. The decision activities inside the factory are analyzed using the GRAI method. The functions of a CIM shell based on AI techniques are explained as a toolbox supporting design and implementation of the intelligent workcell controller.

1. FACTORY AUTOMATION MAIN AIMS

The battle around decreasing throughput times, reduced stocks, and orders exactly on schedule, is presently fought at four fields mainly driven by the availability of ever increasing computing power and the advent of symbolic data processing (AI Artificial Intelligence or AIP Advanced Information Processing). Main aims in these fields are:

- a) Flexible Manufacturing:
constantly changing range of products.
- b) Intelligent Data Handling:
instead of storing answers to production problems, store methods to generate answers when needed.
- c) Flexible Short Term Planning:
making production philosophy practical on the shop floor.
- d) Integrated Organization:
CIM - moving data not paper.

This paper deals with areas a, c, d and the impact of AIP technologies in reaching these goals. We describe a methodology of analyzing and designing the heart of a CIM system, the intelligent workcell controller, and report about a proto-type implementation in the electronics industry (car radio production). The workcell controller is a software package which bridges the gap between CAL Computer Aided Logistics and CAM Computer Aided Manufacturing (fig. 1).

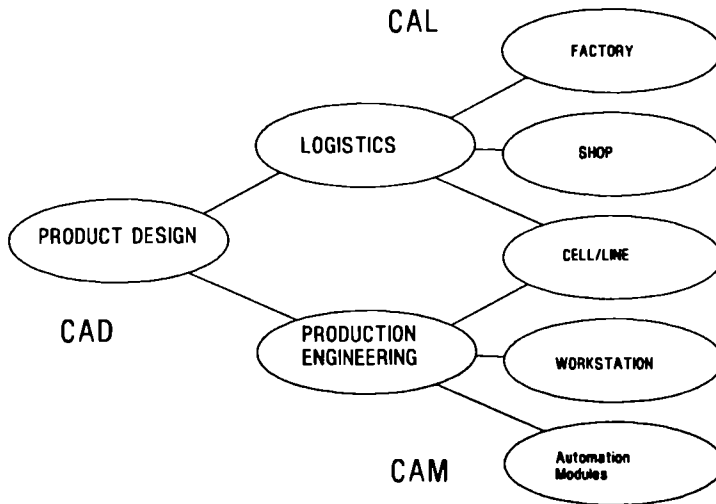


FIGURE 1
Conceptual view for a CIM architecture.

2. TASK DOMAIN ANALYSIS

2.1 General characteristics of CIM systems

Using the definitions of Systems Theory [1], CIM systems are

- open systems
- hierarchical systems in structure and processes
- systems with adaptive and maintenance mechanisms to answer to external variations and to keep the obtained equilibrium
- systems to which there is not a best way to achieve a given objective, in opposite to the closed systems to which there is a cause effect relationship
- systems with several conflicting objectives or constraints expressing a network of relationships among system variables.

Reality has answered to this situation by installing an Information Network and a Decision Network on top of the physical Plant with its product flow, material flow and factory layout. We concentrate on the Decision Network, or the Management System, in the following. This is the domain of OR Operations Research as a discipline, and of AI Artificial Intelligence as a (new) software technique.

2.2 Production strategy

Before thinking about the software means for implementation (either procedural languages like FORTRAN etc., or symbolic processing as in AI languages), the production strategy has to be clearly decided upon. In industries like electronics, tyres and cables, or cars we find the following production characteristics:

- assembly operations mixed with semi-continous flow type processes (soldering, extrusion, vulcanization, painting)
- large variation in batch sizes
- large variety of products
- machine flexibility (more than one machine can be used for any process)

- operation on a product)
- sequence dependent set-ups
- queuing by process.

A number of production strategies for controlling the manufacturing have evolved which can best be classified by batch size and number of product variants (fig. 2). The main problem in reality, however, is how to deal with unexpected events in manufacturing. In this respect, especially the MRP-approach

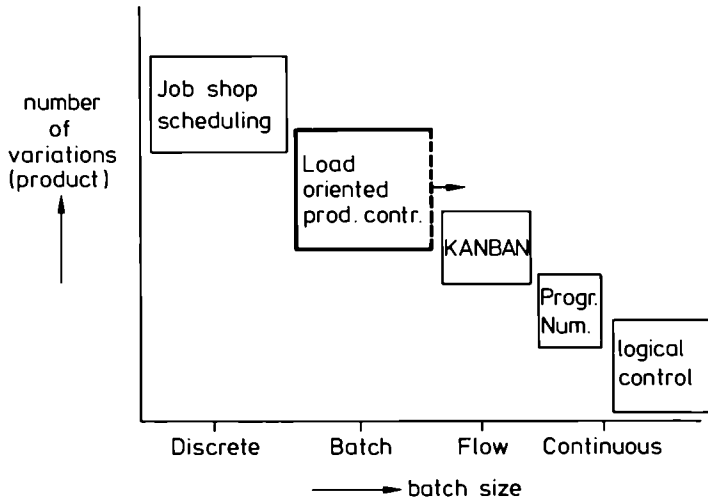


FIGURE 2
Range of applicability for different production strategies depending upon batch size and number of product variants [1].

has proven its inflexibility. This has led to mixed strategies trying to optimize between the top-down MRP and bottom-up KANBAN approach (fig. 3). The four

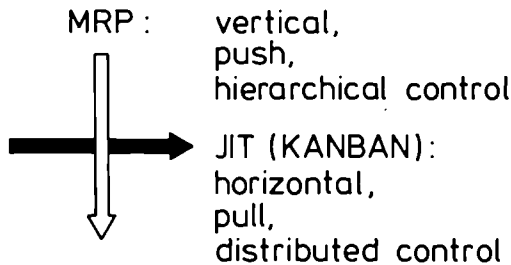


FIGURE 3
Search for the optimum: push - pull, hierarchical - distributed, topdown - bottom-up.

competing strategies mainly differ in their need for data accuracy they have to rely on, and for production disturbances they have to cope with:

- a) MRP Manufacturing Resource Planning
 - Contingencies do not happen (high data accuracy necessary)
- b) KANBAN Just-in-time
 - Contingencies must not happen (zero defect anticipated)
- c) OPT Optimized Production Technique
 - Contingencies are attacked by combining MRP and JIT (bottleneck scheduling applied)
- d) CIM 932, Computer Integrated Manufacturing
 - Contingencies are coped with by adding flexibility to OPT through real-time feedback from shop floor to logistics department (AI methods applied).

CIM 932 is an acronym for the AI approach being favoured in this paper and under development in ESPRIT Project 932 'Knowledge based realtime supervision in CIM' [3].

Fig. 4 shows a practical example of the CIM 932 production strategy applied to a car radio factory. The production process starts with the insertion of SMD surface mounted devices into printed boards and finishes with automated assembly and test of the complete radio including the cassette tape drive. The radio is separated into three groups of parts which are running through the factory obeying different production strategies:

- The Main Print acts as triggering device, is customer order driven and is pushed through the factory.
- The JIT (just-in-time) Assemblies are triggered by the main print and pulled from their (small) KANBAN buffers.
- The Sub Assemblies contain critical parts like ASIC's with long delivery times and have to be planned.

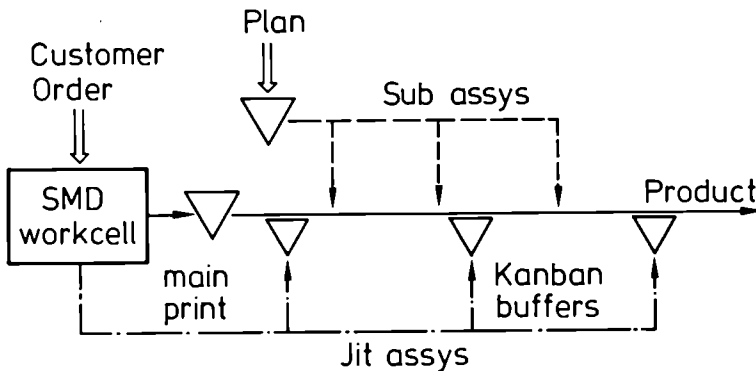


FIGURE 4
Product, production and production strategy as divided into three parts: order driven, planning driven and KANBAN pulled (example car radio production).

Main goal of the Design-for-Production effort within CIM is to minimize the subassembly part of the product and to maximize the JIT part (tools: product modelling, group technology).

Main goal of the Dynamic Scheduling effort within CIM is to find the optimum trade-off for

- throughput time short
- orders exactly on schedule
- stocks small
- capacities well utilized.

According to a recently developed closed theory, called workload oriented production control [2], the main parameters to control and supervise the manufacturing process are:

- average stocks
- average and spread of throughput time
- order priority rules
- capacity time course.

Main graphical tool is the workcell throughput diagram (fig. 5) which is used on-line for planning, diagnosis, and control. In such highly responsive production systems,

- quality of products and plant equipment is prevention-based rather than inspection-driven;
- no time-gaps exist between the planning and operation function in the factory; i.e. adjustment of plans to the actual state of the plant is done in real-time by continuous information feed-back.

The brain to control and optimize this process is the workcell controller. The software technology applied to implement the workcell controller is called AI Artificial Intelligence.

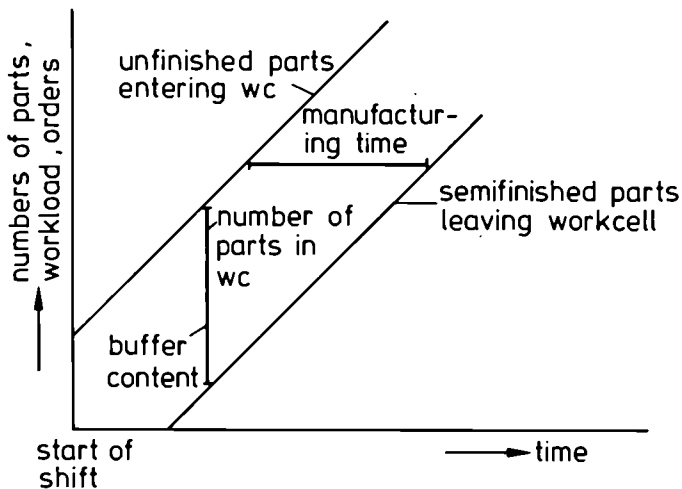


FIGURE 5
Workcell throughput diagram (principle) [2].

3. WHY USE AI?

Basic task in AI is:

Solving Optimization Problems
Under Uncertainty by
Processing Current Information.

This classical definition of Learning well describes the situation on the factory floor: Interpretation, Diagnosis, Planning involve decision making with numerous options and factors, are partly logic, partly intuition, use heuristic rules, and are not suitable for conventional algorithmic techniques.

The concept of the intelligent workcell controller corroborates several AI disciplines like distributed decision making, reasoning and planning, signal understanding and data interpretation. As an intelligent controller, it must reason about its own actions, i.e. has control, explanation and learning capabilities, organized e.g. via blackboards [4]. Learning in the restricted case of factory operation means: Updating of knowledge bases during the inference process in real-time. Real-time depends on the hierarchical level under consideration and ranges from minutes to hours in case of the workcell controller.

Optimization of the factory is basically nothing else as operations on constraints. First step is: Devide planning tasks into subproblems which are (partly) independent and can be solved separately (constraint posting [5]). The GRAI decision network analysis method supports this separation. Second step is: Resolve conflicting goals, or make their influence quantitative (e.g. multi-criteria decision making [6]), or establish quantitative trade-offs between them by either

- knowledge based simulation [7], or by
- understanding the relations between constraints that can provide an effective basis for focussing the plan generation activity: conflict resolution by integrating multiple perspectives [8].

The modern production strategy described above which is the optimized trade-off between the top-down planning approach (MRP Material Requirement Planning) and the bottom-up or data driven behaviour of the shop floor, directly mirrors onto the AI terminology of hierarchical planning and heterarchical or opportunistic planning where subplans are generated independently [9].

4. AI-FEATURES AND TOOLS

4.1 Knowledge representation

AI or symbolic processing uses all kinds of information, including uncertain and intuitive knowledge, as a basis for decision making or reasoning. These types of knowledge are:

- a) Factual Knowledge
($1 + 1 = 2$)
- b) Declarative Knowledge
(all humans are mortal)
- c) Heuristic Knowledge, Experience
(if ... then)

- d) Inferentiell Knowledge
(the knowledge that is available
after application of a) - c))
- e) Meta Knowledge
(knowledge about knowledge).

Formalisms to represent these knowledge types are:

- a) logical or declarative
($a \rightarrow b, b \rightarrow c \Rightarrow a \rightarrow c$)
- b) procedural
(LISP)
- c) object-oriented
(frames)
- d) rule-based
(production rules)
- e) netlike
(semantic nets)
- f) and all kinds of mixings. Systems which use mixings are called hybrid representation systems.

Especially OOP Object Oriented Programming potentially is the programming style of the future. It clearly separates data from procedures and supports modular programming, thus facilitating maintenance, extension and implementation of software products into existing environments. Its main features are:

- generic and individual objects
- hierarchy and inheritance
- characteristics data in slots
 - attributes (values)
 - behavioural description (methods)
- message passing.

The application of OOP to factory simulation is detailed in [3].

4.2 Representation languages

In implementing the knowledge representations, different layers can be distinguished:

- L0 Assembler: the set of solvable problems is maximal
- L1 LISP, Prolog, C, Pascal: conventional languages
- L2 OPS5, KEE, ART: general purpose tools
- L3 SIMKIT: special purpose simulation tool
- L4 CIM Shell: application specific tools
- Ln: set of solvable problems is smallest.

In this definition, a CIM shell contains a model of the factory environment, empty knowledge bases, a versatile user interface for rapid prototyping and operation, inference mechanisms for reasoning, and tools for analysis, design

and implementation of CIM software modules.

From the list above it is obvious, too, that AI and CIM shells can be (and are) implemented in well known languages like C, and not only in the so-called AI-languages LISP, PROLOG and related dialects.

4.3 Expert systems

Expert systems (XPS) are the most widespread AI application at present [10] [11]. They give advice to any level of experienced or unexperienced user in a wide range of applications like diagnosis, optimization and decision making, based on heuristic knowledge of experts. The most powerful approach to problem solving of this kind is MBR Model Based Reasoning, also adopted in the CIM shell described below. Main features of MBR are:

- It leads to deep knowledge about a system (other than shallow as in rule-based XPS).
- The XPS has an explicite computational model of the structure, behaviour and the principles of a system.

A rapidly increasing number of XPS shells is available on the market (see app.). Some of them, e.g. SAVOIR, are a good means to enter the field of AI by implementing a special case and to experience the limits and possibilities of knowledge engineering (learning by doing). Others are large hybrid tools supporting general AI methods and applications, e.g. Knowledge Craft, ART, KEE, S1.

4.4 Blackboard systems

As mentioned already in chapter 3, blackboard systems are especially well adapted to factory applications as they support decision making starting from a hierarchical planning model. E.g., the BBl software [12] is a design tool for intelligent systems (an intelligent system is capable of reasoning about its actions to improve its performance in achieving a goal). Basic concepts of BBl are:

- a) The functionality of BBl will in particular be developed towards the control of reasoning, explanation and learning.
- b) The BBl system allows the explicit and interpretable representation of task specific and domain knowledge about actions, events, states and relationships between the latter.

Task specific knowledge is represented as a framework, e.g. AKKORD for assembly tasks. In case of AKKORD domain specific knowledge may be the knowledge about the construction of a site layout [13].

- c) The design principle is based on a knowledge abstraction hierarchy which consists of an architecture, a framework and an application. The related software engineering principle is that of a modular and layered design [14].

The architecture which is also referred to as the BBl kernel is a set of basic knowledge structures which represent all actions, states, events and facts in the system. This level is independent of the problem class (e.g. arrangement problems), the problem solving method (e.g. assembly method) and the application domain (e.g. construction site lay-out) [12].

- d) Main constraint on the construction of different levels of the abstraction hierarchy is that there exists a uniform standard of knowledge content and representation at each level of the hierarchy. This has the advantage that

new applications can be defined by configuring and augmenting existing knowledge modules. The efforts are towards an Open Systems Interconnection capability [15].

The great potential of blackboard systems lies in their sophisticated and flexible control of reasoning:

The explicitly represented layered approach of solving problems through steps of refinement in BBl is more powerful and closer to the human reasoning than the forward or backward chaining mechanisms in hybrid tools like KEE. Conflict resolution by e.g. least premise complexity or weights assigned to rules is just a very weak representation scheme compared to the rating mechanism in BBl. This is more valid if a planning framework such as AKKORD is available.

Furthermore, the constraint satisfaction mechanism in BBl is typical for planning problems as it can be described as reducing a set of alternatives by applying constraints. In production planning this could e.g. mean that at the beginning of the reasoning the time a shop order shall be executed is not fixed during a day. But through the application of constraints such as availability of resources the order is restricted to a specific time period. An example for such a constraint satisfaction system for the factory environment is the OPIS system [16].

A detailed evaluation of a blackboard system for factory use can be found in [17].

4.5 Target systems

Until now, most large expert systems including the WC-controller are being developed on sophisticated LISP machines like Symbolics and LMI. Workstations like DEC, SUN are attacking the AI market segment now both as development and target machines. Though poor in LISP performance, an increasing number of C-implimentations of expert systems and AI tools support the trend towards multi-purpose workstations and PC's as target systems. The software trend towards run-time versions of the large tools as KEE, Knowledge Craft, ART which download knowledge bases and graphics but not the complete development environment, point to the same direction. However, the field of AI target systems is quickly changing.

5. CIM SHELL

5.1 Implementation methodology

Heart of the CIM shell is the methodology of implementing expert systems at the factory floor, and the factory model as such. The shell supports

- decision network analysis
- decision network design
- factory simulation
- controller architecture
- controller design
- controller implementation
- down loading of code kernel (run-time version, target system).

Parts of the shell are described in detail elsewhere [3].

5.2 Decision network

The decision network acquisition and design tool is based on the GRAI method [18]. Basic concepts of GRAI are:

- a) General structure of production systems
 - decision system
 - information system
 - physical system
- b) Hierarchical structure of the decision system
 - levels defined according to planning horizon and planning cycle
 - GRAI-Grid as graphical tool to represent interrelations from a top-down view
- c) GRAI-networks for detailed modelling of decision activities.

Main idea is to structure the production management system hierarchically according to the time horizon and -periods of the decisions to be taken. The hierarchical model is improved towards distributed control by defining the range of freedom (or responsibility) for the lower levels expressed in terms of objectives, constraints, decision variables and decision rules. Fig. 6 gives an example of the GRAI design grid for the car radio factory, fig. 7 a refined view of an activity center based on a description language which has similarities to Petri-Nets, and which allows the detailed modellization by hierarchical decomposition of decision or executive activities. Fig. 7 exhibits three decision activities: these are the places where to install expert systems for advice giving and, later, for direct feedback control. The GRAI-net formalism turned out to be a valid tool for knowledge acquisition regarding hierarchical planning, subgoal interaction, and constraint interdependency.

5.3 Factory reference model and controller architecture

The factory reference model of hierarchical control serves as a common language between AI researchers, OR (Operational Research) people and plant personal to reveal problems, differences and similarities between different application sites and technical disciplines. The model, based on [19], and originally developed [20] for robot control, is mapped onto the GRAI grid thus combining the concept of time horizons and planning periods with the hierarchical control levels of the factory (fig. 8).

Each box in fig. 8 contains a controller to perform these basic tasks:

- goal or task decomposition from upper to lower level
- sensory data processing
- world modelling.

The controllers, in this view, have the task to bridge the planning gap between logistics and the CAM area (figs. 9-12). In detail, each controller especially on the shop- and workcell level is split up into the same structure (fig. 13) with several submodules each representing an expert system (XPS) for solving special control tasks. The expert systems are grouped around the conventional planning module (Plan Module) which in some implementations is part of a real-time production planning system (PPS), but in many factories it is still missing at least below shop-level. The expert systems, in this respect, simply have the task to refine the microplanning on the corresponding hierarchical layer by considering the real situation of the area under control.

The controller requirements specifications are detailed in [3].

FCT. H/P	SALES	PLANNING	PROCURE- MENT	MAINTENANCE	RESOURCE MGT.	PERSON. MGT.
4 YEARS 4 MON.		4 Years plan Manufacturing resource masterplan				
1 YEAR 4 MON.	Forecasts qty/fam. /month	General Master Plan types of equ. installation qty of pers. qty/fam./m.	Definition of vendor- capacities qty/part/m. Update parts classes	Masterplan for Maintenance	Equipment management policy	Personnel policies
6 MON. 1 MON.	Forecasts qty/fam. /week	Master Plan resources capacities qty./fam. /week	Orders for common & overplanned parts Orders for ext. Subasys qty/part/week		Work preparation	Forecast capacities Training programs
2 WEEKS 1 WEEK	Orders priorities /customer qty/type /day	M.R.P. availability of material & oth. res. qty/type/week fresh. Termin	Short term orders for spec. parts Urgent parts Internal Subasys qty/part/day	Planning maintenance	Planning of modifications or new equipm. or tools	Plan allocation of personnel extra pers. people/hour
3 DAYS 1 DAY		Release orders availability of material & oth. res. M./Type/Datum	Reserve parts & Subasys qty/parts /order	Planning of maintenance activities	Reservation of equipm. off line set up	Allocate personnel JIT<=>Sub people /workcell
3 SHIFTS REALTIME		Sequence orders Call parts Trigger JIT-producer	Stores output	Diagnose first aid repair release of repaired equipm. Machines	Set up	Adjust personnel- allocation

FIGURE 6
GRAI grid. Double arrows: decision frames; simple arrows: information paths.

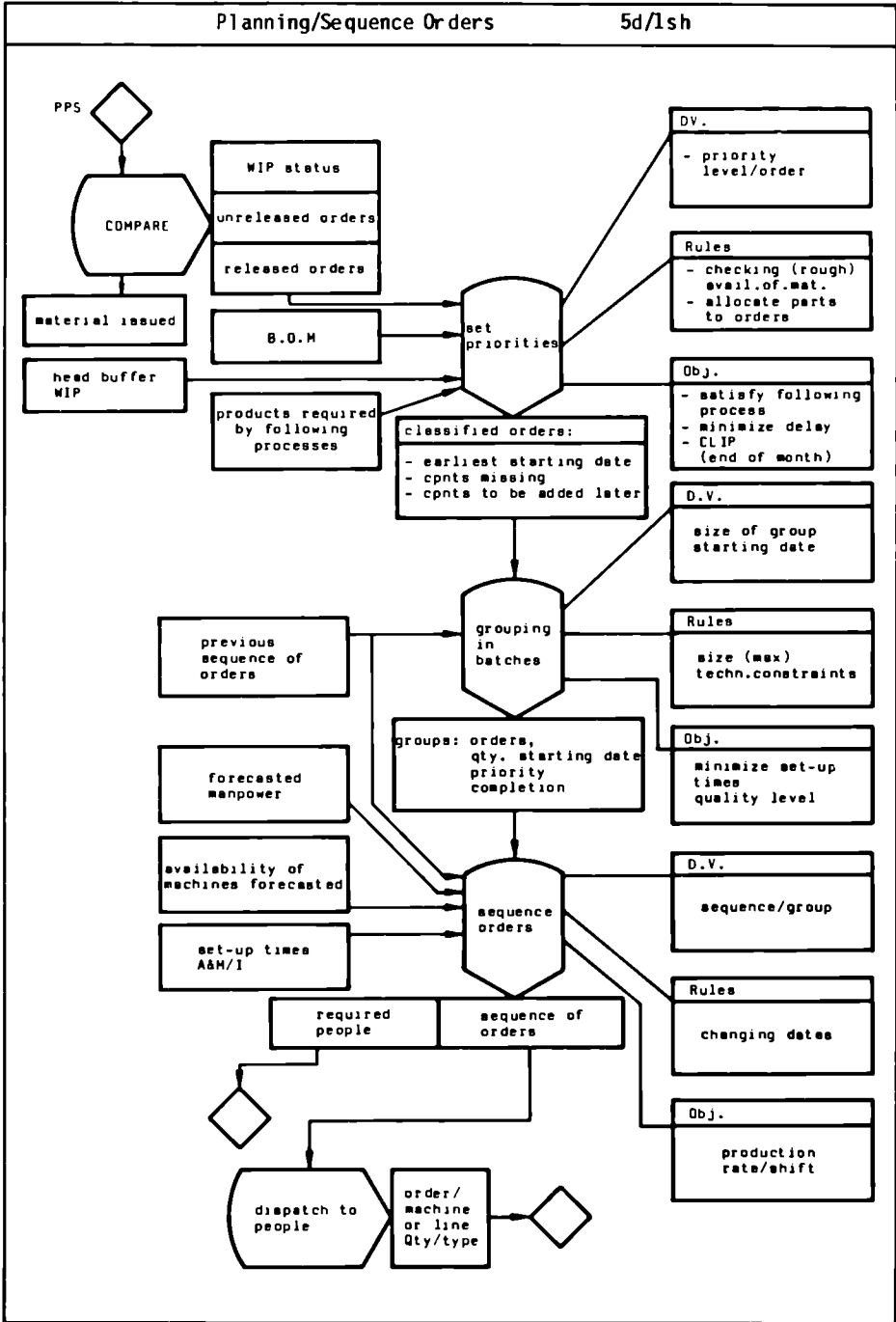


FIGURE 7

GRAI description of activity center 'Planning and sequencing of orders'. See decisional activities with decision variables (DV), rules and objectives (Obj).

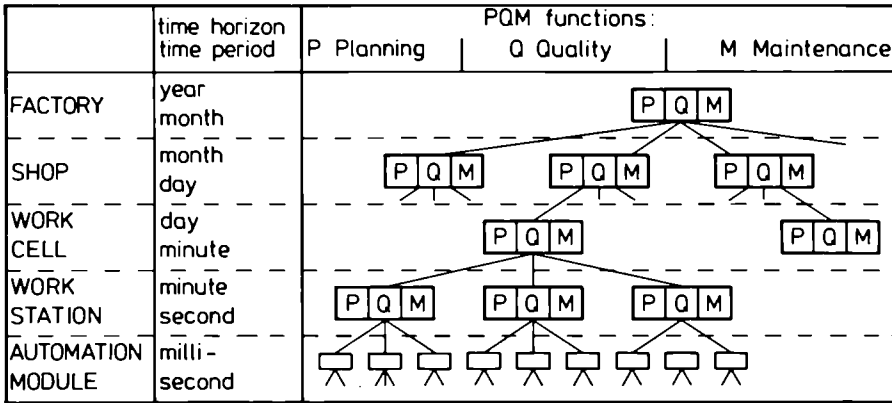


FIGURE 8
Factory reference model.

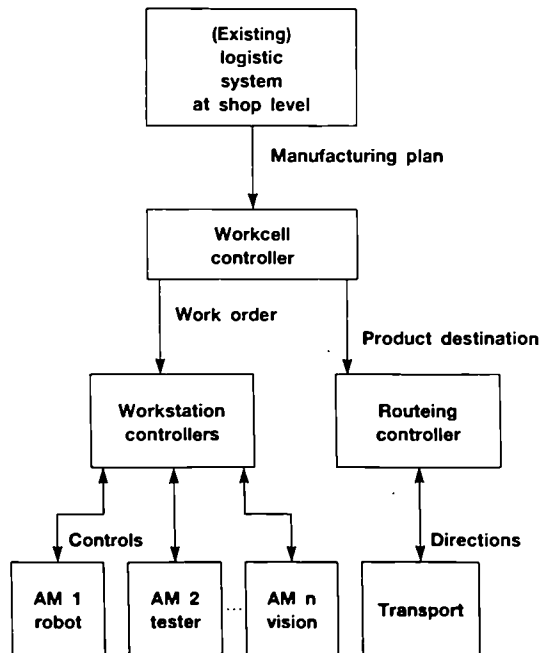
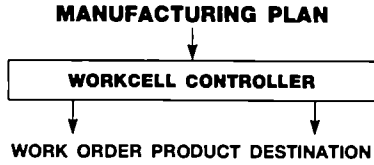
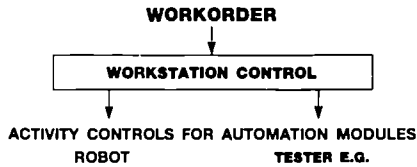


FIGURE 9
Datagram for controllers at workcell and workstation level.



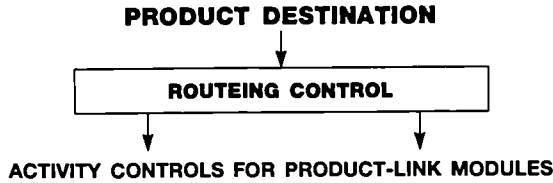
- Input** : A manufacturing plan for a certain time period (day, week)
- Function** : - Distribute workorders over the workstations and transport system to fulfill the manufacturing plan.
 - Use production strategy
 - JustInTime production
 - Push production
 - Pull production
 - Push / pull
 - Optimization algorithms
 - Corrective actions if planning cannot be completed because of workstation failure
- Output** : - Workorder commands for workstations
 - Routing commands for the transport system
- Feedback** : - Performance of workstations
 - Performance of transport
 - Product movement, tools e.g.

FIGURE 10
 Input-output functions of workcell controller.



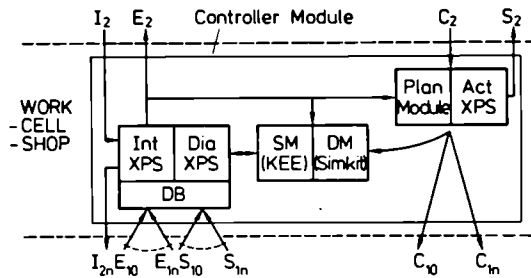
- Input** : A workorder for a workstation
- Function** : - Coordinate the activities of the automation modules (Robot, Tester, ...) to fulfill the workorder (I.E. assemble A)
 - Use a coordination strategy
 - Optimization algorithms
 - Convert the workstation commands to commands which can be interpreted by the diversity of automation modules
 - Provide an operator interface
- Output** : - Activity controls for automation modules
 - Status information for workcell controller
- Feedback** : - Performance of automation modules

FIGURE 11
 Input-output functions of workstation controller.



- Input** : A request to send a product to a workstation
- Function** : - Route product to its destination
- Use a coordination strategy
 - Distributed control algorithms
 - Central control algorithms
 - Provide an operator interface to define the transport network
- Output** : Direction controls for product-link modules
- Status information for workcell controller
- Feedback** : - Performance of product-link modules

FIGURE 12
Input-output functions of routing controller.



- C₂ Command from upper level
- I₂ Information about environment from upper level
- S₂ Status report about plan execution to upper level
- E₂ Interpreted status of equipment to upper level
- C₁₀-C_{1n} Commands to lower level
- S₁₀-S_{1n} Status signals concerning command execution from lower level
- E₁₀-E_{1n} Status signals from lower level equipment
- DB Real time data base
- SM Static world model
- DM Simulation of dynamic production flow

FIGURE 13
Controller-structure: Interpretation XPS, Diagnostic XPS, Action Planning XPS, conventional planning module.

6. CONTROLLER REALIZATION

The controller is the most important part of the production strategy implementation (fig. 4). A first prototype controller is being realized for the first SMD Surface Mounted Device workcell at the Philips car radio factory AFW Wetzlar/FRG. The controller supports the two lowest levels of the planning function (fig. 6) with the optimization activities (fig. 7).

- queue configuration (set priorities),
- grouping in batches, and
- order sequencing.

The planning problem is as follows:

- 6000 - number of printed circuit boards per day
 - 8 - typical number of different orders per day from the PPS-Plan
 - 50 - number of possible different product types
 - 12 - number of machines
 - 3 - number of identical machines.

This means more than 6000 billion possible schedules (combinatorial explosion) which have to be handled by the foreman being responsible for this group of machines. Today, a high level PPS Production Planning System generates a production plan weekly and with a three weeks horizon. The microplanning is done daily and has to consider the actually available resources such as personnel, machines and raw material. The high complexity of this task is further complicated by unreliable data:

- a) The PPS-Plan is mainly based on orders from customers and not sufficiently optimized for the available capacities on the shop floor.
- b) There must be a minimization of set-up times so that specific prints have to be assembled in consecutive batches which are different for each sub-workcell.
- c) There exist changes to the PPS-Plan due to unforeseen high priority customer orders.
- d) The capacity of the workcell is difficult to predict.

The foreman tries to solve this problem by applying heuristic rules which cut down the range of possibilities. But it is very difficult to later judge the actions based on these rules in order to optimize the production process because they are not explicitly defined nor is their application structured. This makes it very difficult to exchange experiences between the foremen to steadily improve the production process as a whole.

The task of expert system technology is to specify explicitly the existing rules in the knowledge engineering process. An example for such a rule is:

Print-order-dispatching-rule:

```

IF
    this print is of type Main-Print
    and has axial-components inserted
    or is the specific Main-Print-1
    and the free-capacity of the SMD-workcell
    is less than 90%
    and the print is in the list of high-priority prints
    and the currently processed print on the SMD-workcell
    is set-up compatible
  
```

THEN

Start immediately processing this-print on the
SMD-Workcell.

In close cooperation with the foremen and the responsables for the PPS-plan such rules have been analyzed to find a general cycle of microplanning as it is performed by the foreman each day. In particular the aim was to make clear any mismatching in the rules used for microplanning and for generating the PPS-plan.

A simplified result for a basic planning cycle has the following steps:

- a) Choose an order from PPS-plan.
Constraints:
 - a) PPS-Plan for this day
 - b) released material
 - c) priority for next workcell
- b) Choose workstation.
Constraints:
 - a) print type
 - b) workload already planned
- c) Build a significant batch at the SMD-Workcell.
Constraints:
 - a) print type
 - b) PPS-Plan for this day
 - c) workload already planned
 - d) workload already produced
 - e) priority of next workcell
 - f) PPS-Plan for next three days
 - g) commonality of prints at different workstations
- d) Repeat steps a, b and c until PPS-Plan for this day is empty.
- e) Check for workstations with free capacity.

Based on this cycle a workcell controller was designed fitting into the factory reference model (fig. 14). The development environment consists of KEE and SIMKIT running on Symbolics LISP machine, as target system planned is DEC's Mikrovax II/GPX workstation featuring KEE's downloaded run-time version.

7. CONCLUSION

The rapid prototyping approach inherent to OOP and AI has proved to be superior both for increasing flexibility in implementing software as well as getting people together for problem specification and knowledge acquisition. The proposed methodology for decision network analysis and the resulting factory reference model based on planning horizons and decision frames as main constraints determining subsystem interaction is modelling factory reality in an acceptable way. The intelligent workcell controller as brain of the production system is starting to prove its usefulness in running factories. Blackboard systems offer advantages over general knowledge engineering environments with regard to flexible control mechanisms and prestructured decision hierarchy. Target system implementations of AI favour the C-language and multipurpose workstations or PC's, eventually with LISP and PROLOG humming boards.

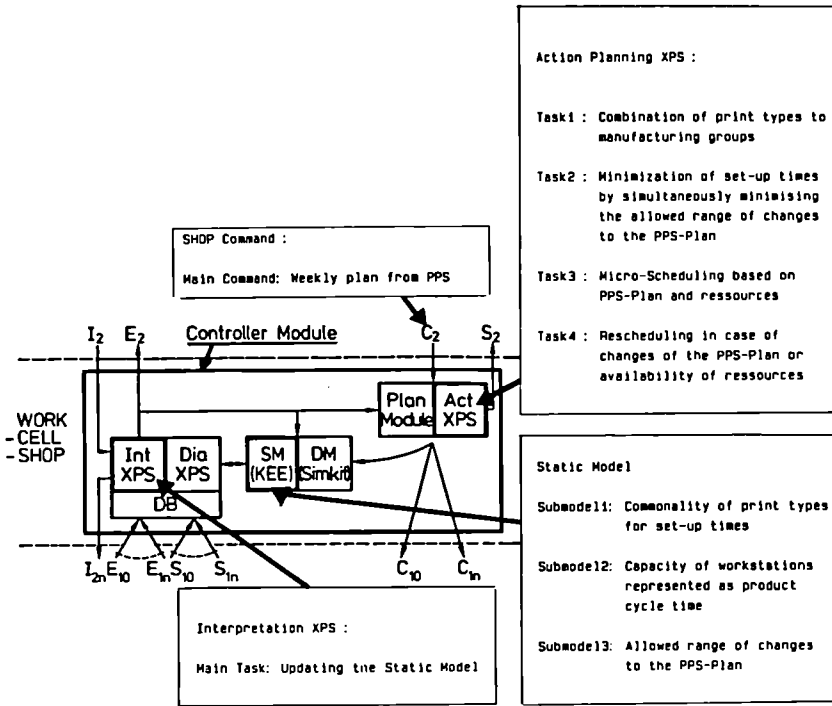


FIGURE 14
Controller modules for SMD workcell controller.

ACKNOWLEDGEMENT

This work was partially funded by the European Community under contract ESPRIT 932. Furthermore, we want to thank Mrs. Dr. Barbara Hayes-Roth and Mike Hewett, Knowledge Systems Laboratory of the Stanford University, and our Philips colleagues from AFW Wetzlar and CFT Eindhoven for active help and fundamental information.

REFERENCES

- [1] Bartalanffy, L., General Systems Theory - A Critical Review, in: Systems Behaviour, Harper and Row Publ., 1972, pp. 29-49.
- [2] Wiendahl, H.-P., Belastungsorientierte Fertigungssteuerung, Carl Hanser Verlag, München 1987.
- [3] Meyer, W., Knowledge based realtime supervision in CIM - The workcell controller, in: ESPRIT '86: RESULTS AND ACHIEVEMENTS, Directorate General XIII (Editors), Elsevier Science Publishers B.V. (North-Holland), 1987, p. 33.
- [4] Hayes-Roth, B., Garvey, A., Vaughan Johnson Jr., M., Hewett, M., A layered environment for reasoning about action. Rept. No. KSL 86-38, Knowledge Systems Lab., Stanford Univ., Cal. 94305, USA.
- [5] Stefic, M., Planning with constraints (MOLGEN: part 1). Artificial Intelligence 16 (1981) 111.
- [6] Fandel, J.S. (ed.), Multicriteria Decision Methods and Applications, Springer Verlag, 1985.
- [7] Ramana Reddy, Y.V., Fox, M.S., Husain, N., The Knowledge-based simulation system, IEEE Software, March 1986, 27.
- [8] Smith, S.F., Ow, P.S., LePape, C., McLaren, B., Muscettola, N., Integrating Multiple Scheduling Perspectives to generate detailed production plans. Proc.: SME Conf. AI in Manufacturing, Sept. 1986, Long Beach, USA.
- [9] Stefic, M., Planning and Meta-Planning (MOLGEN: part 2). Artificial Intelligence 16 (1981) 141.
- [10] Buchanan, B.G., Expert Systems: working systems and the research literature. Expert Systems 3 (1986) 32.
- [11] Pan, L.F., Survey of expert systems for fault detection, test generation and maintenance. Expert Systems 3 (1986) 100.
- [12] Hayes-Roth, B., A blackboard architecture for control, Artificial Intelligence Journal 26 (1985) 251.
- [13] Tommelein, I.D., Johnson, Jr. M.V., Hayes-Roth, B., Levitt, R.E., Sight-plan: A Blackboard Expert System for Construction Site Layout, IFIP WG5.2 Conf. Expert Systems in Computer-Aided Design, 1987, February, Sydney, Australia.
- [14] Goos, G., Hartmanis, J. (eds.), Distributed systems - Architecture and Implementation, Plenum Press, New York 1982.
- [15] Zimmermann, H., A standard layer model, in: Paul E. Green (editor), Computer network architecture and protocols, Plenum Press, New York 1982.
- [16] Smith, S.F., Ow, P.S., The use of multiple problem decompositions in time-constrained planning tasks. Proc.: 9th Intl. Joint Conf. AI, Los Angeles, August 1985.
- [17] Isenberg, R., Comparison of BBI and KEE for building a production planning expert system, in: Proc. 3rd Intl. Exp. Systems Conf., London, June 2-4, 1987, pp. 407-421.
- [18] Doumeingths, G., Methodology to design CIM and control of manufacturing units, in: Methods and tools for CIM Lecture notes in computer science 168, p. 194, Springer Verlag, Berlin 1984.
- [19] Mesarovic, M.D. et al., Theory of hierarchical multilevel systems, Academic Press Inc., NY and London 1970.
- [20] Albus, J. et al., Theory and practice of hierarchical control, IEEE 1981, Comcon Fall, National Bureau of Standards, Washington, DC 20234.

APPENDIX: Inventory of commercial expert system
development tools

REMARKS: Noncomplete, no guaranty for pricing,
market rapidly changing, situation
from january 1987.

MAC = Apple Macintosh
PC = IBM PC or compatible
WS = Workstations (e.g. SUN, APOLLO)
VAX = VAX
IBM = IBM mainframe
LISP = LISP machines (e.g. Symbolics, LMI)
OTHERS = Other machines

Expert system tool	MAC	PC	WS	VAX	IBM	LISP	OTHERS
1st Class		x					
Superfile ACLS		x					x
Acquaint (Daisy)		x					
Analyzer plus		x					
APES		x		x			x
Arity ESDP		x					
ART			x			x	
Class		x	x	x	x		
CognitIF			x	x	x		
Crystal		x					
Deja Vu		x					
Dexpert		x	x	x	x		x
Duck			x	x		x	
Envisage				x			
ERS		x					
ESE (ESDE&ESCE)					x		
ESP adviser		x		x			x
ESP Frame Engine		x					
Experfacts	x						
Experkit	x	x					
ExperOPS5	x	x					
Expert 2		x					
Expert 4		x					
Experteach		x					
Expertease		x					x
Expertedge (TESS)		x					
Exsys		x					
Extran-7		x	x	x	x		x
Golem		x					
Guru		x		x			
Humble		x					x
In ate	x			x		x	
IKE						x	

Expert System tool	MAC	PC	WS	VAX	IBM	LISP	OTHERS
Inference manager (AL/X)		x					
Insight I, II & III		x		x			
Intelligence/Compiler		x					
Intelligence Service		x		x			
Iroise				x			x
K.1					x		
KDS		x					
KEE			x			x	
KES		x	x	x			
KIP	x	x	x	x		x	x
KISS		x					
Knowledge Craft (SRL+)				x		x	
Knowledge Workbench			x				
Knowol		x					
L'expert		x					
M.1		x					
Macexpert	x						
MP-LRO & Le_Lisp			x	x			x
Nexpert	x	x		x			
Omega						x	
OP55 (+)		x	x	x			x
OPS83		x	x	x			x
PARSEC	x	x	x				
Personal Consultant Plus		x					x
Picon						x	
Poplog			x	x			
Qtime		x					
Reveal		x					
Rulemaster		x	x	x			x
S.1		(x)	x	x		x	
Savoir		x	x	x	x		x
Series PC		x					
Service Shell	x	x	x	x		x	x
Small-X		x					
Snark		x			x		
Super Expert		x					
TIMM		x		x			
Topsi		x					x
Trouble shooter		x					
Turbo expert		x					
Tvaice							x
Vie Ket						x	
VP Expert		x					
Wizdom		x					
Xi		x					
Xsys		x					
Zexpert							x

Project No. 179

VLSI AND REAL TIME CONTROL OF PLANT AUTOMATION SYSTEMS

Paul R. Doree

GEC Research Ltd., East Lane, WEMBLEY, HA9 7PP, U.K.

Abstract

In recent years, computer integrated manufacturing (CIM) technology has been experiencing difficulties of implementation not originally envisaged. Present research has been unable to exploit existing processor hardware in terms of the low cost and high performance and accuracy that is required for real time controllers. There is an increasing requirement for control systems to be responsive in real time to the changes normally found in real world environments. This is particularly the case where the control system is expected to work faster and under more difficult conditions. Such conditions can only be overcome by the development of novel VLSI processor architectures for real time control. The present work under ESPRIT Project 179 has concentrated upon the development of three VLSI integrated circuits that will result in increased performance and reliability as well as reducing control system size and cost. The applicability of these circuits for real time control is considered. Part of the work has involved the development of a design methodology for future electronics systems.

1. INTRODUCTION

The control and automation of production plants is a very complex task. Today it is solved by multi-processor systems using existing hardware with well defined functions assigned to the various processors. In addition to the technical complexity of the control task, time constraints frequently have to be observed in order to achieve satisfactory control system performance. This is particularly the case for closed loop motion and position controllers employing external sensors.

It is the objective of ESPRIT Project 179 to apply very large scale integration (VLSI) techniques to systems for plant automation. The work has involved identifying functions within control systems which can be profitably mapped onto custom VLSI circuits to improve control system performance. In the first year of the project, an analysis of a representative robot control system was performed to identify functional areas for integration [1]. It was soon realized that the real time requirements of high throughput and non-real time user could only be satisfied by novel VLSI architectures. However the development of a specialized circuit may result in a substantial subsystem cost decrease. Examples of this are in motor drive interfaces and two of the integrated circuits address this requirement.

A supplementary objective of this project is the development of a design methodology which supports the design of a control systems addressed within the

project. The design methodology is currently being based on experience gained during the circuit definition and has highlighted some of the decisions taken during the design process. The design methodology also includes the development of a block structured language for future control system design. This language will allow the formulation of specifications for software and VLSI circuits in a way that is more familiar to control system design engineers.

2. CIRCUIT DEVELOPMENT

Three integrated circuits are being developed. Two of these circuits are Current Controller ICs and are being implemented to reduce subsystem costs. The third circuit, a VLSI Matrix Coprocessor represents a more generalized solution to the computational bottleneck situation of existing real time control systems particularly in the area of coordinate transformation, multivariable axis control, and external sensor signal processing. It is intended to use the matrix coprocessor to facilitate the application of advanced real time control strategies within the axis control and the state observation of the controlled mechanical system.

2.1 THE CURRENT CONTROLLER CIRCUITS

Two Current Controller Integrated Circuits are being developed for the motor drive interfaces. The first of these circuits, the Current Controller for DC servo drives has been implemented in bipolar silicon technology and functions to the original specification. The second Current Controller IC for DC brushless drives is still in development. The specification of the first Current Controller IC is discussed as follows.

Many robots are driven with DC motors, the DC voltage being supplied by transistor bridge circuits. The power range for these drive types varies from 500 watts to 10kW. The velocity and/or position control loops of the drives usually contain an internal current loop which is normally scaled for an input voltage of 10 volts DC.

The Current Controller which is illustrated in figure 1 generates pulses in order to switch the power transistors rapidly on and off which are arranged in a bridge configuration to guarantee the current flow through a DC motor in both directions. The actual average voltage across the motor is dependent on the duration time that the transistors are switched on. This technique of constant pulse frequency with variable "on time" is called "pulse-width-modulation".

The Current Controller IC contains all the necessary circuitry for generating the four pulsed waveforms. In addition the IC contains circuitry that monitors the output waveforms in real time. If any malfunction occurs then all the outputs are immediately zeroed to protect the power transistors and the motor from damage. The resulting chip is fabricated in double layer metal Integrated Injection Logic bipolar technology. The chip size is 17 square millimetres and contains analogue and digital logic. The chip supply voltage is 15v and has a power dissipation of 300mW.

3. MATRIX COPROCESSOR

3.1 Computational Problems in Robotics

A single VLSI coprocessor chip known as the Matrix Coprocessor is being developed to enhance the computational capability of real time control systems. Before discussing the coprocessor chip it is worth reviewing some of the

computational problems in real time control systems. This field includes the field of robotics as well as state estimation systems. Robots play an important role in industrial plant automation, their usefulness depending on the accurate and rapid control of manipulators. As robots become more advanced they are expected to work faster and under more difficult conditions, requiring not only an increase in the execution speed of existing algorithms but the use of more sophisticated and computationally demanding requirements for robot control.

For example, in kinematics we are interested in the geometry of robot arm motion with respect to a fixed reference coordinate system. In a serial robot, the position and orientation of a manipulator can be found by a coordinate transformation given the joint coordinates [2]. A more useful, and more difficult problem is to solve the Inverse Kinematics problem; ie given the desired position and orientation of the manipulator, find the necessary joint coordinates by matrix solving. Inverse Kinematics is essential to some of the more advanced control algorithms such as those developed for paint spraying robots [3].

Accurate dynamic robot control requires the real time computation of joint forces in addition to kinematics. Dynamic equations are often expressed in vector (state space) form and hence require the use of vector (and matrix) arithmetic. Each computation involves a large number of multiplications and additions [4]. The Kalman Filter is a powerful method for real time control of noisy systems, and has been shown to be particularly useful for the control of the radial axis of a robot [5], also makes extensive use of matrix arithmetic.

Since future systems will require sampling rates of greater than 100Hz [3], and 300Hz is typical for the degree of freedom manipulators, it is quite evident that real time control requires now a very powerful means of matrix and vector computation using single precision floating point arithmetic (32-bit). At present advanced real time robot control is only possible using large expensive floating point vector

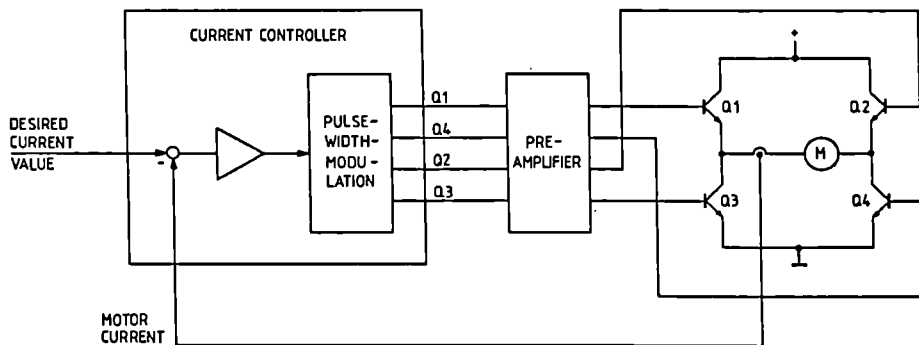


FIGURE 1 CURRENT CONTROLLER FOR DC SERVO DRIVES

processors. Economically viable use of these algorithms needs the provision of a small, but powerful in terms of high throughput and non-real time user interaction, unit based on an advanced real time VLSI circuit.

3.2 Coordinate Transformations

The CORDIC technique was first introduced by Volder in 1959 as a special purpose digital computer (Coordinate Rotation Digital Computer) for real time air-borne computations. The scheme geometrically rotates a three-dimensional vector onto the $x - y$ or $x - z$ plane in a Circular, Linear, or Hyperbolic trajectory as shown in figure 2. Volder only considered the circular case but suggested that multiplication, division, conversion between binary and mixed radix systems, extraction of square roots, hyperbolic coordinate transformations, tan, tanh and generation of logarithms could be performed.

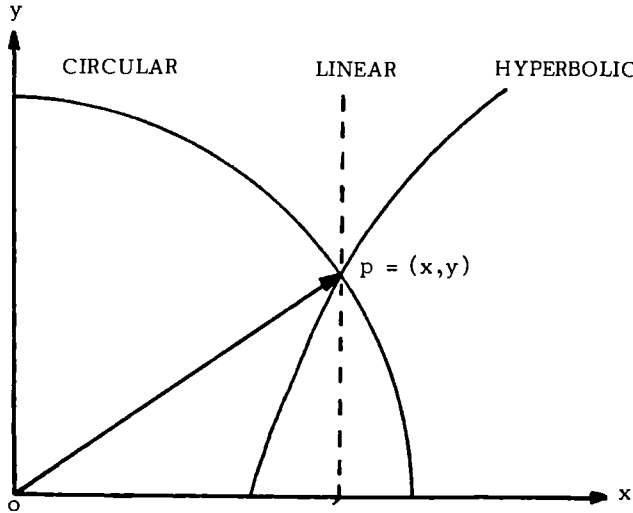


FIGURE 2 CORDIC TRANSFORMATIONS

Walther unified the algorithm so that all of the above elementary functions could be generated by a suitable set of input parameters.

The Linear transformation involves arithmetic operations such as addition, subtraction, multiplication, and division. The Circular transformation involves trigonometric operations such as sines, cosines, arctangents and computes metric distance. The Hyperbolic transformations involves operations such as sinh, cosh, arctanh and square roots. From these functions, tan, tanh, exponentials and natural logarithms can be formed as shown in figure 3.

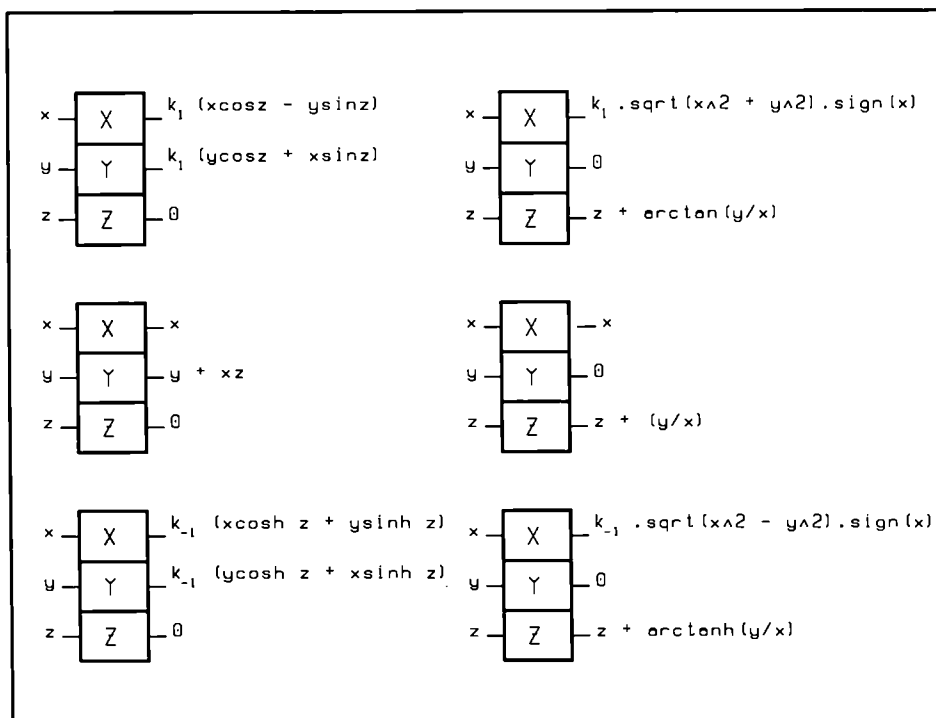


FIGURE 3 CORDIC FUNCTIONS

The CORDIC technique is iterative and is described as follows. Let the vector $P(xyz)$ exist in the three dimensional x , y , and z planes where x and y are Euclidean and z is polar. Let the vector P be transformed such that after the $(i+1)$ th iteration, the new vector position is given by;

$$P(i+1) = (x(i+1), y(i+1), z(i+1)) \quad (1)$$

where

$$\begin{aligned} x(i+1) &= x(i) + y(i) \cdot s(i) \\ y(i+1) &= y(i) - x(i) \cdot s(i) \\ z(i+1) &= z(i) + a(i) \end{aligned}$$

and $a(i) = \arctan(s(i))$ where $s(i)$ is arbitrary. If the plane containing vector P is rotated onto the $x - y$ plane then the elementary functions obtained are Euclidean. This is achieved by choosing values of $a(i)$ such that $z(i)$ is forced to zero. If the plane containing the vector P is rotated onto the $x - z$ plane, then polar values result. This is achieved by choosing $s(i) = z^{-p}$ where p is a suitable sequence of integers. Then the CORDIC operations are simple additions, subtractions, shift and compare steps. this ensures that the CORDIC scheme is exponentially convergent since $s(i) \rightarrow 0$ as $i \rightarrow \infty$.

The elementary functions generated after n iterations are shown in figure 3. The resulting precision depends on n , K_1 , and K_{-1} where K_1 and K_{-1} are scaling

factors. The sequence of constants for $a(i)$ and $s(i)$ are stored in ROM. Walther analysed the range of inputs under which the scheme converges and the convergence domain, and showed that n iterations produced an accuracy to n bits.

The circular transformation expressed in matrix form is;

$$\begin{bmatrix} \cos z & -\sin z \\ \sin z & \cos z \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

The form (2) is encountered in robotics [9], image processing and graphics. The Coordinate Transform technique has been used in generating the kinematic and dynamic solutions of robotic systems and is well structured. Instead of explicitly computing the sines and cosines and then forming a matrix-vector product, the CORDIC hardware will rotate a vector about an angle in a single operation. This is considerably faster than computing the matrix-vector product and only requires sufficient storage for the initial position and the angle of rotation.

3.3 MATRIX SOLVING

3.3.1 Currently Available VLSI Building Blocks

Before considering the proposed matrix coprocessor, it is useful to examine currently available arithmetic products in relation to real time computational requirements. A number of integrated circuit manufacturers now sell high performance arithmetic products such as multipliers and multiplier-accumulators, however these in general use fixed point arithmetic. However in the last couple of years a number of single and double precision floating point arithmetic units have come onto the market. Although currently available floating point units are capable of very high computation rates, they are necessarily appropriate for matrix operations. Even the provision of pipelined registers, which supports digital filtering well, cannot overcome the I/O bandwidth problem inherent in matrix computations. Although matrix computations are suitable for pipelined computation, the large numbers of data transfers required between a conventional arithmetic unit and memory effectively diminish the computation rate. This necessitates the development of a processing unit that is more advanced than those currently available. Such a processing unit must have an architecture that closely matches the matrix operand accesses that are required.

3.3.2 Review of Possible Algorithms and Architectures for a Matrix Coprocessor System

The recent advent of VLSI microelectronic technology has created new architectural horizons enabling the implementation of large scale matrix/vector computations directly in hardware. In the context of VLSI it has become necessary to distinguish between an algorithm and an implementing structure. The primary goal of the architecture was to support reasonable efficient implementation of a very broad variety of matrix algorithms. A number of processor realizations were considered, which ranged from single processors to various implementations of parallel array processors. In addition a number of matrix algorithms that are required for real time control were studied to assess the ease of mapping onto a particular architecture. A number of parallel architectures were studied and these are discussed very briefly as follows.

3.3.2.1 Parallel Processor Types

In recent years, VLSI architectures for highly parallel computations have been extensively investigated. The following is a list of parallel processor types considered.

1. Systolic Arrays
2. The Two Dimensional Wavefront Processor [10]
3. Systems using Bit Serial and Bit Parallel multipliers
4. Matrix Partitioning using the Hwang Algorithms [11]
5. SIMD arrays of processing elements
6. MIMD arrays of processors

The alternative to array architectures is the single chip processor.

The I/O bottleneck in VLSI systems can present serious restrictions on algorithm design. The challenge is to design algorithms that can be partitioned such that the amount of communication between modules is as small as possible. Further in many of the array processor architectures, each processing element (PE) would be a single chip, and some systems would require large arrays of several chip types. Such a system would not be cost effective for robot or real time controllers. the alternative is a single chip processor that is driven by a host processor as a peripheral. Such a processor would be cost effective for such control systems. The specification for such a processor is outlined as follows.

3.4 Matrix Coprocessor Architecture

The Matrix Coprocessor (MCP) is a unique special purpose device optimised for matrix/vector based computations and Coordinate Transformations. There is no known equivalent device currently available on the market. The MCP combines VLSI architecture with an optimised instruction set and efficient matrix computation algorithms to produce a high performance chip. The MCP operates as a stand alone device with host processor support for control and instructions.

The coprocessor functions either as a matrix/vector solver or as a coordinate transformer and provides a set of matrix primitives that can be used to formulate most matrix vector algorithms. The MCP also provides two explicit vector operations, a coordinate transform primitive which is based on the CORDIC technique and vector cross product. The MCP is capable of performing matrix operations on any size matrix or vector up to (32 x 32). The coprocessor system is designed to operate with the minimum of interference from the host processor. Coprocessor instructions are written in blocks involving a number of matrix operations, these blocks are downloaded by the host into the coprocessor local memory. The coprocessor only needs to interrupt the host processor when a new block of instructions is required or when an error has been detected. hence the coprocessor system has a low interrupt to compute time ratio. in this way, matrix operations are relegated to background activities which frees the host for other functions such as general housekeeping, data acquisition, peripheral control and computation sequencing.

The MCP architecture supports single precision 32-bit floating point arithmetic which is partially compatible with the IEEE P754 standard. Data words are 32-bit and a dual 32-bit bus is used internally to increase throughput. The dual bus allows an external 32-bit multiplier to operate at full pipeline speed. Typical target performance ranges from 3 - 20 Million Floating Point Operations per second (MFLOPs) depending on the operation. The MCP instructions are decoded internally and sequenced on chip.

3.4.1 MCP System Architecture

The Matrix Coprocessor system is comprised of the MCP chip, a 32-bit Floating Point Multiplier, up to 128K of local memory which is split equally into 64K for instructions and 64K for data, and a host interface. The organization of these components is illustrated in figure 4.

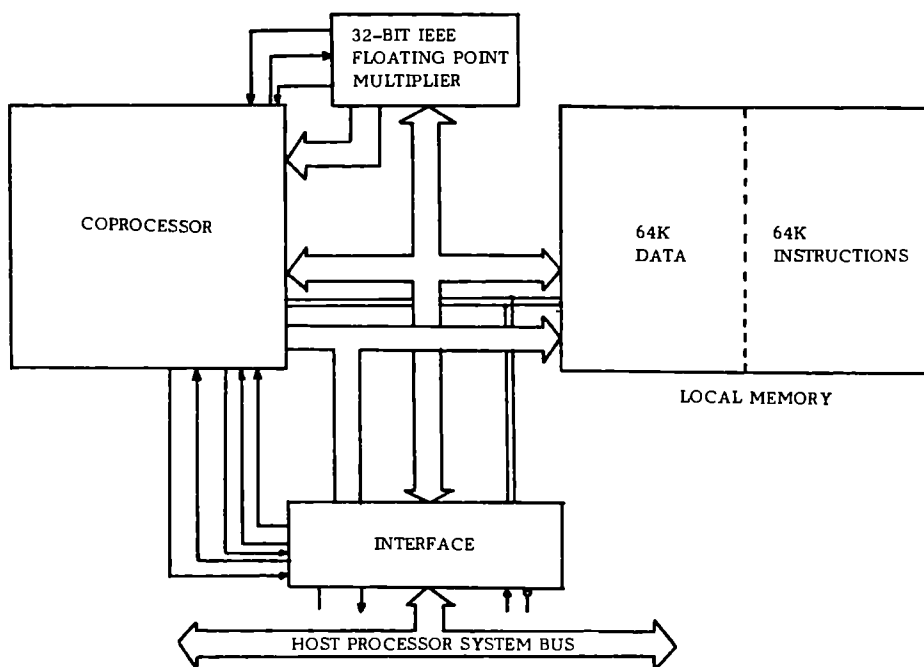


FIGURE 4 MCP SYSTEM ARCHITECTURE

The matrix/vector computations are compiled into the MCP local memory in blocks which are either sent via the interface or are burnt into PROM when the computation is iterative. Each block can be thought of as a subroutine, and consists of instructions and data which may represent the whole or part of an application model. For a control system application model, one block could represent the computations required to implement a Kalman Filter. The number of such blocks which can be stored is limited by the available local memory space. Algorithms

which exceed the available space can be computed using overlay techniques in which new blocks are downloaded over existing blocks. The distribution of computations between blocks is performed at compilation (taking into account available data and instruction memory) and is transparent to the user.

3.4.2 MCP Hardware Design Methodology

With all system simulation it is necessary to demonstrate before fabrication that the system performs the correct function, and that the system timing is correct. If the system is initially designed and simulated with both a behavioural and functional simulator then the architecture and function of the system and its component chips will be known before implementation. The component chips will have been simulated in some detail to ascertain functional correctness and the simulations logged.

The programming language C was chosen for the architectural simulator for this circuit design. During this design phase major new architectural features were studied. From this study, detailed requirements were prepared to determine the feasibility of implementing each feature within the timescales of the project. Extensive verification tests against worked examples took place during this phase. Once the system architecture was checked and verified it was then further decomposed into modules and the detailed function and interfaces of each module was specified.

The hardware description language ELLA™ was used to specify the MCP system functionality. The architectural features were divided into modules and the detailed function and interfaces of each module specified. This decomposition was done in parallel with a geometrical analysis of the internal structure of the MCP chip. From this a floorplan was generated. The timing of the whole MCP system was specified and a provisional "path analysis" performed to determine that the data and control signal propagation between blocks was achievable within the specification. When a new function was added to the ELLA™ module, a minimum set of tests would be run to check that the function was performing correctly. Then the function was exercised against boundary conditions.

Gradually the MCP chip was assembled from these module descriptions. Initially the MCP chip consisted of some twelve module descriptions, but in parallel with the floorplanning exercise, this was reduced to nine parent modules. Behavioural descriptions were written using ELLA™ for the local memory and the 32-bit IEEE floating point multiplier which then enabled a complete description of the MCP system to be assembled. The current description of the MCP system in ELLA™ amounts to some 6000 lines.

3.4.2.1 The Hardware Verification Process

An matrix/vector algorithm was described by the assembly language and assembled into instruction sequences that will run on either the architectural or functional models. The flow of events is shown in figure 5. The instruction sequences will run on the C model, but must be converted into 'ELLA™ instructions' to run on the hardware description model. Data must also be converted from real into IEEE format. The stages of the ELLA™ simulation model what would happen with real chips. At the start of a run, the model is initialized. The local memory is loaded with instructions and data, and the MCP chip started. The MCP chip then fetches, decodes, and executes instructions just like the real hardware. After a successful

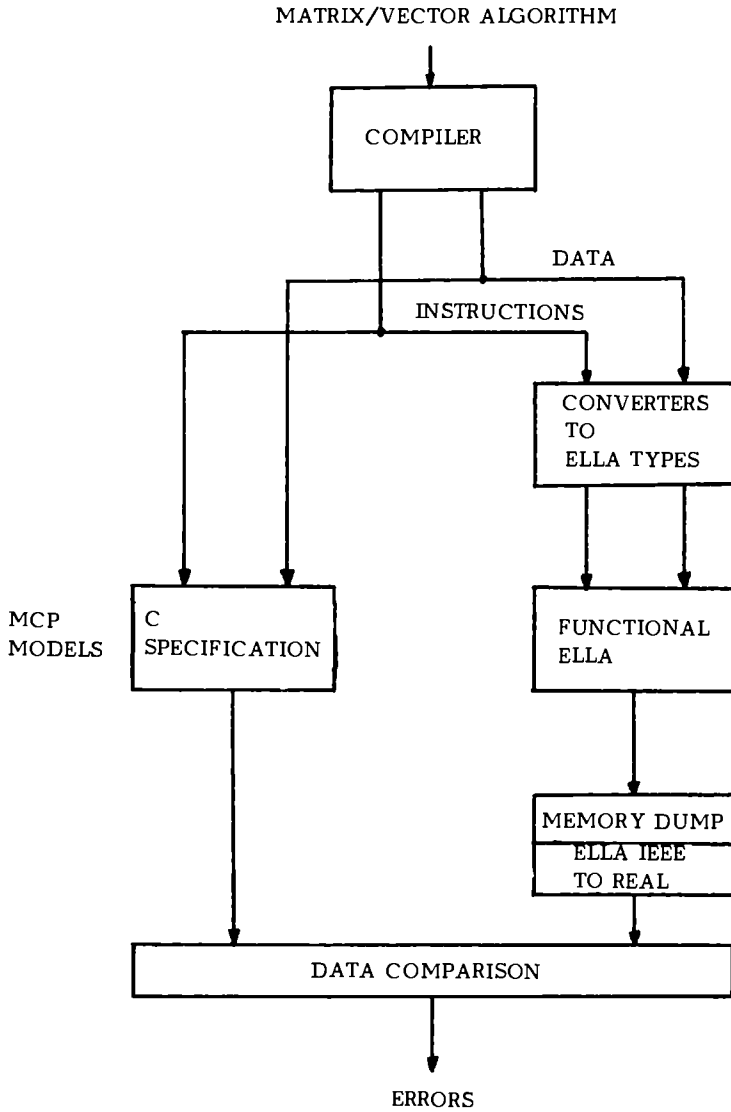


FIGURE 5 VERIFICATION OF THE MCP FUNCTIONAL MODEL

run, the result is converted from ELLA™ types back into real data, and the MCP status register can then be read by the user. Once the ELLA™ model is working the design was implemented on a silicon compiler using the GENESIL™ software.

For implementation on the silicon compiler the MCP chip design was partitioned into nine parent modules as was done for the ELLA™ description. The definition of these modules was constructed bottom up from a library of generic block primitives. Gradually the MCP chip was assembled from this database of leaf descriptions. Behavioural descriptions were written in GENIE™ for the local memory and the 32-bit IEEE multiplier which enabled a simple MCP system to be assembled. The parent modules and the MCP chip/system was tested as follows.

Simple algorithms or instruction primitives were run on the functional ELLA™ model and the actions of a particular module/block were monitored. The changes in inputs and outputs for this module/block were recorded in a log file. In-house software was written for parsing the ELLA™ log files to produce test vectors for the testing the layout modules. Simple verification of the module/block functions was provided by including expected outputs in the test vector files for a given set of inputs. The test vectors were then run on the layout modules/blocks with the GENESIL™ software performing automatic verification of the module/block functions with the results written either to the terminal screen or to a trace file. In this way the MCP chip could be ascertained to work correctly to function before committing to silicon.

A version 0 compiler has been written to simplify the writing of matrix/vector control algorithms.

3.5 Matrix Coprocessor Applications

The MCP is a single coprocessor which it is hoped will have wide application in the fields of real time control and graphics. One suitable algorithm is the adaptive on-line control of a paint spraying robot. An algorithm for this purpose has been developed within the project. The algorithm computes the settings of the robot servos necessary to position the spray gun correctly with respect to the object being sprayed. Repeated application of this algorithm ensures that the robot adapts to changes in the position of the object. Accurate tracking of the object requires the algorithm to be evaluated at least 100 times per second for a practical application. So to achieve this a wireframe graphics display of a paint spraying robot has been constructed. The robot and the object being shown are displayed on a terminal screen. The position of the object is controlled by a mouse, and the object and the wire frame robot will move in real time. This demonstrator will allow the performance of the MCP system to be evaluated. It is intended to show using the demonstrator software the performance enhancement possible utilising the MCP against more conventional processor/numeric coprocessor matrix techniques.

3.6 Matrix Coprocessor Expansion Card

An MCP expansion card (MCPC) is being developed for the MCP in order to demonstrate the MCP capabilities. Using an IBM PC/AT compatible machine as HOST platform, the MCP system will be sited on an expansion card (MCPC).

3.7 Layout of the Matrix Coprocessor Chip

The MCP chip layout is currently being implemented on a Silicon Compiler, using the GENESIL™ developed and marketed by Silicon Compiler Systems Incorporated, San Jose, California in the USA. A silicon compiler is of great interest in a systems design environment in that allows an engineer familiar with the requirements for system performance to design directly onto silicon. Use of a silicon compiler enables rapid prototyping of an integrated circuit design. Design descriptions are entered at functional block level using built-in circuit primitives that the compiler software supplies. A block definition is then compiled for speed and area and then simulated enabling speedy changes if a block is not within specification or does not function correctly.

The GENESIL™ software imposes a strict limitation on the clocking structure of a circuit. This two phase clocking strategy simplifies the task of achieving correctness by construction by avoiding circuit races which simplifies the process of design verification. A silicon compiler also allows future design upgrades as smaller geometry processes permit the inclusion of more functions onto a single piece of silicon. It is inconceivable that the Matrix Coprocessor system would be fabricated onto a single piece of silicon, but future process developments may allow the implementation of the present MCP chip and the floating point multiplier onto a single piece of silicon.

The current MCP chip design is targeted to be 12 millimetres square, contains just under 100K transistors, and will be fabricated in 2um Double Level Metal Bulk CMOS technology.

The Matrix Coprocessor has an architecture optimised for matrix computations and coordinate transformation and has application in real time control systems, graphics systems, or as a matrix accelerator in mainframe computers. The architectural challenge has been to find a configuration that is computationally efficient, small enough to be producible, and flexible to find wide application.

4. DESIGN METHODOLOGY

The objective is to be able to design control systems for plant automation systems which are part of CIM systems. VLSI technology has been advancing very rapidly in recent years. Not only has the number of transistors per chip increased but the ability to design integrated circuits has been extended. For many years systems designers have designed their hardware using off-the-shelf components, however now it is possible for the system designer to specify an integrated circuit to meet his own requirements.

It is quite clear that a major problem will be to fully exploit the possibilities of the IC development in a system design. The ability to specify and design ICs must be available to system engineers. Currently gate array and cell library techniques are widely used for mapping board level designs into ICs. However for real time control systems, the full potential of IC performance can only be realised by mapping functional solutions directly into silicon.

During the course of the project a number of control system studies have been undertaken to identify suitable subsystems for integrating as VLSI circuits. In the case of the Matrix Coprocessor the specification was only reached after a long

period of interaction between IC designers and control system engineers. Clearly it is desirable to accelerate this specification phase.

At present a number of computer packages can be used to specify designs at the system and software levels. Currently a variety of Computer Aided Design (CAD) packages are available for IC design. One possible goal of the design methodology is the development of design information that is common to the systems designers and the IC engineers.

4.1 Block Structure Language

In the future several Computer Aided Engineering (CAE) systems will be involved in the design and maintenance of an overall plant automation system. The centre of the design activities is usually a user specific CAE system which performs such tasks as requirements definition, project management and system integration. Future design tasks of specific devices such as VLSI circuits for real time controllers would be supported by specific CAE systems dedicated to these tasks.

Under the influence of MAP the International Standardisation activities have accelerated in recent years. A distinct tendency has been observed in the move towards describing the functional description of control systems using function block oriented descriptions.

The objective under this part of the project has been to define a syntactical method for the representation of function block structures with a view to introducing this method into international standards. Part of the documentation resulting from this phase of the work has been used as an input to the EIA as a proposal for the extension of EIA Draft Standard RS 511.

5. CONCLUSION

Circuits which will be key components in future real time systems have been specified. Of these circuits, the current controller has been fabricated and works to specification. The specification for the Matrix Coprocessor IC was reached after a long interaction between IC designers and control system engineers. Whilst it is realised that real time control is not all current controllers or matrices and vectors, the research performed in specifying the chips has formed the groundwork for real time products. As CAD systems improve and reduce IC design cycles then developments such as arising from the Design Methodology will it is hoped shorten the route between control system requirements and IC developments. This is especially important now that custom IC design is becoming readily available to system designers.

6. REFERENCES

1. Potton, S.L. and Klittich, M., Integrated Microelectronic Subsystems for Plant Automation in: Esprit '84 (North-Holland, Amsterdam, 1985)
2. Potton, S.L. Study of robot kinematics and dynamics, Esprit Pilot Project 5.2, Report No. GEC 14/2 (1983).
3. Reeves, P.J., Algorithms for the Adaptive On-Line Control of Paint Spraying Robots, Esprit Pilot Project 5.2, Report No. GEC 24/1 (1984).

4. Vickers, M.J., A Comparative Assessment of Techniques for Forming the Dynamics Equations of Parallel Topology, Esprit Pilot Project 5.2, Report No. GEC 22/1 (1984).
5. Parry, G.A., Control of the Radial Axis of a Robot Using Kalman Techniques, Esprit Pilot Project 5.2, Report No. GEC 23/1 (1984).
6. Volder, J.E., The CORDIC trigonometric computing technique, IRE Trans Elect Comp Ec-8 No. 3 (July 1959) 330-385.
7. Walther, J.S., A Unified Algorithm for Elementary Functions, Proc. Joint Spring Computer Conf. (July 1971) 379-385.
8. Spaniol, O., Computer Arithmetic Logic and Design (John Wiley & Sons, 1981).
9. Paul, R.P., Robot Manipulators: mathematics, programming and control (MIT Press, 1981).
10. Kung, S.Y., Arun, K.S., Gal-ezer, R.J. and Basker Rao, D.V., Wavefront Array Processor: Language, Architecture and Applications, IEEE Trans. Comput. C-31 (1982) 1054-1066.
11. Hwang, K. and Cheung, Y.H., Partitioned Matrix Algorithms for VLSI Arithmetic Systems IEEE Trans. Comput, C-31 (1982) 1215-1224.

ELLA™ (ELECTronic Logic LAnguage) is a trade mark of the Secretary of State for Defense. ELLA is available from Praxis Systems plc, 20 Manvers St., Bath, UK.

GENESIL™ and GENIE™ are trade marks of Silicon Compiler Systems Inc. of San Jose, USA.

Project No. 278

AN INTEGRATED SENSOR-BASED ROBOT SYSTEM USING TACTILE AND VISUAL SENSING

N. Ghani, Z.G. Rzepczynski, P.M. Hage, A.E. Adams

MARI Advanced Microelectronics Ltd, 32 Grainger Park Road, Newcastle Upon Tyne. NE4 8RY

This paper presents preliminary results of an Esprit project which is concerned with developing and integrating multiple advanced sensor systems into a robot system aimed at handling workpieces in an unstructured environment. Two types of gripper-mounted tactile image sensors have been developed and integrated into a specially designed modular gripper system. The tactile sensing role has been implemented through an image processing system which provides spatial correction data during grasping. Advanced vision algorithms and hardware have been developed and integrated successfully with an industrial robot system. Integration of tactile sensing into this vision-guided robot system has been demonstrated.

1. INTRODUCTION

This paper reports on the progress achieved after two and a half years of a three-year project entitled "Tactile Sensing for Integrated Sensor-Based Robot Systems". This project, which started in January 1985 within the CIM area of ESPRIT, is concerned with the problems of developing advanced sensor systems and integrating them into a practical industrial robot environment. Particular emphasis was placed on the development of a gripper-mounted tactile image sensing system, and investigation of the role of such a sensor when deployed in cooperation with an advanced vision system and other sensors [1]. The problem of handling randomly presented workpieces in an unstructured environment [2], was selected as the application vehicle for the technology development of the project.

The partners in this collaborative project and their roles are as follows:

1. MARI Advanced Microelectronics Limited, Newcastle upon Tyne, UK, (prime contractor, and tactile system developer).
2. Vickers Plc., Joyce Loebel Division, Gateshead, UK, (vision system developer).
3. Fraunhofer-Institut fuer Produktionstechnik und Automatisierung, Stuttgart, FRG, (gripper developer, systems integration and application engineers).

4. Robert Bosch GmbH, Erbach, FRG, (robot controller system developer)
5. University of Newcastle upon Tyne, UK, (specialist support in sensors, VLSI and control).
6. National Technical University of Athens, Greece, (advanced control techniques).
7. New University of Lisbon, Portugal, (investigation of CAD integration issues).

During the first year, a major thrust of the project was to set up two experimental test sites, the primary one at IPA, and a subsidiary one at Joyce Loebel. Each test site started off with standard commercially available equipment: a Kuka 160/15 robot controlled by a Bosch Rho2 controller interfaced to a Joyce Loebel Magiscan 2A vision system. In parallel, research and development work on the non-standard components needed began, and the first prototypes appeared during the second year. The task of integrating and testing these prototypes then began in earnest.

The project is structured into 6 major work packages:

1. Tactile
2. Vision
3. Gripper
4. Control
5. Workpiece Handling
6. CAD/CAM Integration

The progress and results achieved by the project consortium are presented below under these headings.

2. TACTILE

The project has demonstrated two tactile sensing techniques both of which have been integrated into the prototype gripper developed by IPA. Algorithms for recognising tactile images and measuring spatial offsets within the gripper have been developed and implemented on a 68000-based Tactile Processor System which has been integrated with the Gripper Management Processor which is also 68000-based.

The MARI piezoresistive sensor [3] provides a 32 x 32 sensor array on 1.27 mm pitch on a pad 15mm thick. The pre-processing electronics has been implemented as a hybridised circuit and has been integrated with the sensor pad. By incorporating the preprocessing functions directly within the sensor pad the physical link between the pad and the tactile processor system is greatly simplified.

An alternative tactile sensor using an optical effect has been developed by the University of Newcastle upon Tyne. This uses a thin silicon rubber membrane stretched over a transparent acrylic plate which is illuminated at its edges; the tactile forces are thus converted into a visual image which is picked up by a CCD array and lens arrangement. The integration of this scheme into a gripper finger with a rotating pad is a significant achievement as it permits the gathering of tactile images whilst an object is being rotated by the gripper pads. Object orientation, position and slippage information is then available to the robot controller during object manipulation [4].

The tactile processor is a 68000-based system running under the pSOS executive, with a TMS320-10 Digital Signal Processor as a slave. The system communicates with the GMP via a serial link. The tactile processor may be requested either to *learn* a pair of tactile images, or to *recognise* against a previously learnt pair. The output of the recognition process is termed a correction matrix which effectively defines the x, y and z-rotation displacement of the recognised images relative to the learnt, with a confidence factor. At present, recognition takes 2-4 seconds, with most of the analysis algorithms coded in Pascal on the 68000, and the DSP performing some of the low-level processing functions.

At the end of the second year, it was possible to demonstrate at the UK test site, the use of the correction matrix to guide the robot to correctly regrasp an object which had been offset after initial learning. This has now been extended at the German test site to the stage where the correction matrix is being used to improve the accuracy of vision-guided grasping.

The Newcastle University team have investigated the use of hardware techniques for pattern matching. This has been embodied in a prototype custom integrated circuit device currently under test. This device uses the "shadow" technique to extract shape vectors directly from a 8 x 8 image plane [5].

3. VISION

The vision work has followed three main threads: hardware, software, and integration. The starting point for this work was Joyce Loebel's existing Magiscan vision system [6].

The major hardware achievement during the second year was the completion and installation of a low-cost industrial development of the Magiscan hardware, based on a PC/XT (and more recently, PC/AT) compatible host with ROM/RAM disk emulation. The initial vision system has since been replaced by two new versions incorporating all the design concepts that have been developed over the first half of the project. One, the IV20, has been completed now and the other, the IV30, will be completed by the end of the project. Both systems plug directly into PC/AT host computers. A hardware convolver has been installed on the IV20 which is now being manufactured as a VLSI device.

Recent activities at the University of Newcastle upon Tyne have concentrated on the VLSI implementation of several vision functions. The main devices developed are for a real-time histogrammer and a real-time convolver. Both these circuits are designed to operate on 256 x 256 images and each design performs a complete operation within a single silicon chip. A further, more specialist design has been produced with a view to the implementation of the shadow technique of parameter derivation from image information. All designs are based on a 3 micron CMOS implementation [7,8,9,10,11].

The output of the software effort is a package called the RVCS (Robot Vision Control System) which at the end of the 2nd year is capable of locating touching components as well as non-touching. The system uses statistical pattern recognition based on up to 13 measurements. Touching components are handled using "angle-distance" functions. A non-linear convolution is used for edge enhancement while histogramming is used to optimise extraction of internal features. The RVCS is currently being extended to resolve overlapping components. The RVCS package also includes an option to teach the system a gripper template so that the pick up area around the component can be checked to ensure that a collision does not occur.

The main thrust of the integration work has been to enable the vision system to be controlled directly by the robot control system during all phases of operation - calibration, training and recognition. This has been achieved with the present robot controller, providing a simplified user interface through the robot control VDU. The system uses a fixed plan view camera to locate components in its field of view, while a robot-mounted camera is used for close-up imaging (at fixed height) for recognition and more accurate location. The operator is able to initiate automatic calibration sequences via the robot control VDU, as well as train and run the system. Interfacing has now been fully integrated with the latest version of robot controller.

4. GRIPPER

The result of the activity in this work package is a working prototype gripper system which is now integrated into the IPA test bed. This system incorporates a number of design improvements over the initial prototype which was completed during the second year of the project. The modular design addresses the requirements for flexibility in form and function which is related to the need for integration of a variety of sensors. The prototype system includes a gripper module with five pairs of manually exchangeable fingers. The gripper module provides electrical drive for the fingers with closed loop position control. The modular design also allows for the addition of a force/moment sensor module between the gripper and the wrist.

Three of the finger pairs include a z-axis rotation capability. In each pair, one of the fingers incorporates a servo motor which drives a circular pad, while the other finger has a free-wheel pad.

The University of Newcastle team have incorporated their optically-based tactile sensor into one of these free-wheeled fingers. The MARI tactile sensors have been mounted on both rotating pads in one of the pairs. On this pair, the non-driven pad has a spring mechanism to return it to its zero position after grasping.

The gripper prototype is controlled by a 68000-based Gripper Management Processor which handles the closed loop control for both axes, as well as the serial communications with the Tactile Processor and the Robot Control System.

5. ROBOT CONTROL

There are two threads to the control work in the project [13]. The industrial thread is concerned with the hardware and software problems of integration of sensor guidance with a multiplicity of intelligent sensors into an industrial grade robot control system (the Bosch rho2). The academic thread is concerned with the development of adaptive control algorithms which could be applied at a practical level to a control system such as the rho2 [12].

Over the first two years, the major effort on the industrial front has been in investigating, specifying and developing a number of enhancements to the rho2 system which are in the final stages of testing. The main hardware enhancement is the addition of an additional 32016-based processor subsystem into the rho2's multiprocessor architecture. This new processor is dedicated to the management of the intelligent sensor subsystems and the gripper subsystem which is treated here as an intelligent peripheral.

The software effort has also been substantial, involving new software for the sensor processor and enhancements to the existing software. The sensor supervision software runs under a specially-developed multi-tasking operating system on the new board, and essentially mediates between the application language interpreter on the main processor and the sensor subsystems. The language (called BAPS, Bosch Application Programming System) has been extended with a number of sensor-related instructions, and the task of translating and transforming between the functions and coordinate systems at this level, and those provided by the sensor supervision software.

The adaptive control work has been carried out by a group at Newcastle University and the National Technical University of Athens, together with Bosch. This has involved linear and non-linear dynamic modelling of the Kuka robot and simulation studies based on those models. A package to provide dedicated graphical output of modelling and control simulation modules on a Tektronix 4205 colour terminal using a VAX computer has been developed. Because of the lack of available data on the robot, it was necessary for Bosch to modify their controller software so that some of the input/output signals could be logged during operation. This has allowed the theoretical developments of PID, Pole-assignment, General predictive self-tuning and model reference adaptive control algorithms [12,13,14,15,16] to be related to the practical environment of the project.

A further software modification is planned which will allow the gain of the overall rho2 control loop to be updated during operation. This will allow a simplified form of adaptive control to be applied using some of the results of the theoretical work, and hence to demonstrate the practical benefits of the approach.

6. WORKPIECE HANDLING

IPA have the task of putting the full system together and implementing the workpiece handling application on it. The main result here is that the end of the 2nd year, the IPA test site is able to use the prototype gripper to pick up and orientate randomly positioned workpieces (obloid valve blocks which may touch each other) making use of the vision system for sensing. Following the successful integration of the tactile sensors, the system is now using the correction matrix information produced to further guide the orientation process.

Of particular interest has been the development and integration of orientation strategies involving the use of the gripper's rotation axis and the need to cope with alternative grip positions (eg. when components are touching). The preliminary results indicate the flexibility strategies to be used.

7. CAD/CAM INTEGRATION

The New University of Lisbon joined the project in March 1986, nearly half-way through the project. Rather than attempting to merge them into the ongoing development work, it was agreed that their most useful role would be to investigate the problems and the issues that would be faced in trying to integrate a sensor-based robot system of the type being developed into a CAD/CAM environment.

The approach taken is a pragmatic one, with effort going into setting up a working environment which includes similar Components to those used elsewhere in the project. Experimental work in interfacing between various software tools in this environment has been carried out to explore the relationships between CAD, robot programming language, simulators, and focussed on the role of knowledge-based tools and techniques [17].

The involvement of the Lisbon group has been fortuitous, as through them, a strong link has been forged with Esprit project 623, which is concerned with issues of implicit and explicit programming of robot cells.

8. CONCLUSIONS

At this stage of the project, the main achievements so far may be summarised:

- Not one but two working tactile image sensors have been demonstrated, both exhibiting innovative approaches to construction to achieve advances in functionality. Patent applications have been filed on both sensors.

- The role of gripper-mounted tactile image sensors within the sensor-based workpiece handling application has been clarified and defined and is being tested. The tactile image processing system necessary for this role has been implemented and demonstrated successfully.
- The role of vision in the system has been developed to an extent where a high degree of integration exists between an industrial robot control system and an advanced vision system capable of locating and recognising a variety of parts which may be touching each other. Experiments with this integrated system in handling parts has helped to highlight the contribution expected of the tactile sensors.
- An advanced gripper design optimised for parts handling and orientation has been demonstrated and integrated with the robot. The design takes into account the needs of sensor integration within the gripper.

The final phase of the project will centre around the integration of the total application demonstration combined with completion and/or enhancement of the various subsystem developments. The degree to (and the style with) which this total integration will be achieved should help to illuminate the problems of realising the objectives of CIM at the lower levels of the automation hierarchy.

ACKNOWLEDGEMENTS

Over the last two years, much hard work by the members of the seven teams has gone into achieving the progress reported here. The degree of cooperation between the teams has been tremendous, and the contributions of the following persons deserve particular mention: S Redman and S Bell (MARI), R Jones (JL), H Lindner, W Glaess and G Jordan (IPA), Dr W Gerke (Bosch), Dr C Allen, Dr J Finch, Dr M Farsi, Dr A.J Harris and Dr H.C Yung (NU), Prof S Tzafestas (NUTA), and Prof A Steiger-Garcao (UNL).

The support, finance and encouragement received over the last two and a half years from the Commission has been invaluable, and in particular the consortium would like to thank Ms P MacConnaill and Dr A Ikonopoulos for their continuing interest and attention to our needs. The relationship that has developed with the project reviewers, Prof U Rembold of the University of Karlsruhe and Mr F Dogliani of DEA SpA, has been beneficial to the project, and their comments and advice are gratefully acknowledged.

REFERENCES

- [1] Ghani, N., Tactile with Vision: An Integrated Approach Robot Sensing, in: Bernold, T. (ed), Artificial Intelligence in Manufacturing (Elsevier, Amsterdam, 1987).

- [2] Raymond, C., Donath, M. and Olson, W.R., Problems of Vision Directed Robots in an Unstructured Parts Handling Environment, Proc 13th Int. Symp. Ind. Robots 2 (1983) 14/1 - 14/18.
- [3] Ghani, N., and Rzepczynski, Z.G., A Tactile Sensing System for Robotics, Int. Conf. on Intell. Auton. Sys., Amsterdam, Dec 1986 (proceeding in preparation).
- [4] Allen, C.R., Harris, A.J., and Glaess, W., An Integrated Smart Tactile Sensor for 2-axis Robot Grippers. SPIE Proceedings on Int. Conf. on Advances in Intelligent Robotics Systems, Nov 1987, Cambridge, USA.
- [5] Farsi, M., Allen, C.R., and Finch, J.W., Robot Control using Intelligent VLSI-based Sensory Feedback, Int. Conf. on Intell. Auton. Sys., Amsterdam, Dec 1986 (proceeding in preparation).
- [6] Robinson, A., and Hage. P., Coordinating Vision and Tactile Sensing with Robotics, Sensor Review, (July 1986) 130-132.
- [7] Thurlbeck, I., Selmane, M.K., Allen, C.R., Adams, A.E. and Yung, H.C. Towards the Integration of a Real Time CMOS Image Histogrammer. Proceedings of the International Conference on Digital Signal Processing, Florence, Italy, Sept 1987.
- [8] Adams, A.E., Allen, C.R., Farsi, M., Finch, J.W., Harris, A.J. and Yung, H.C. An Integrated Sensor, intelligent Robot for Flexible Manufacturing Systems. Proceedings of the 8th Int. Conf. on Assembly Automation, IFS, 1987, pp.227-236.
- [9] Yung, H.C., Allen, C.R., Finch, J.W., Harris, A.J., Adams, A.E. and Farsi, M. The Development of an Intelligent Sensor-based Industrial Robotic System. IEEE proc. Int. Workshop on Industrial Applications of MVMI, Feb 1987, pp.280-284.
- [10] Allen, C.R., Finch, J.W. and Shepherd, A.J. The Application of Shadow analysis to Tactile Sensors. Advances in Image Processing and Pattern Recognition, North-Holland, 1986, pp.298-302.
- [11] Yung, H.C., The Synthesis of VLSI Signal Processors - theory and example. To be published as a chapter in 'Theoretical Aspects of VLSI Designs', Academic Press, 1987.
- [12] Tzafestas, S.G., Integrated Sensor-Based Robot System, 25th IEEE CDC, Athens (Dec 86).
- [13] Tzafestas, S.G. and Stassinopoulos, G.I., A Decentralised Robot Control Scheme with Self-Tuning PID Controllers, AFCET/IASTED Int'l Symp. on Robots and Art. Intell, Toulouse, June 1986.

- [14] Tzafestas, S.G. and Stavracakis, G., Model Reference Adaptive Control of Industrial Robots with Actuator Dynamics, IFACS/IFIP/IMACS Symp. on Theory of Robots, Vienna, Dec 1986.
- [15] Farsi, M., Finch, J.W., Warwick, K., Tzafestas, S.G. and Stassinopoulos, G., Simplified PID Self-tuning Controller for Robotic Manipulators. Proc. of the 25th IEEE Conference on Decision and Control, Athens, 1986, pp.1886-1887.
- [16] Farsi, M. and Karam, K. Self-tuning and Adaptive Control, Oxford, 1987, pp.7/3-7/10.
- [17] Steiger-Garcia, A., and Camarinha-Matos, L.M., A Knowledge-Based Approach for Multisensory Integration, Nato Workshop on "Languages for Sensor-Based Control in Robotics, Castel. Pasc., Italy, Sept 1986.

Project No. 623

PLANNING AND PROGRAMMING OF ROBOT INTEGRATED
PRODUCTION CELLS.

Spur et. al.

Fraunhofer Institut für Produktionsanlagen und
Konstruktionstechnik (IPK) , Pascalstr. 8-9, D-1000
D - 1000 Berlin 10

In this paper the objectives and the concept for the execution of the R + D Project ESPRIT 623 are outlined. The research activities for computerized planning and programming of robot integrated production cells and the main results reached in the different areas are presented.

1. INTRODUCTION

The overall objective of the project is the realization of prototype systems to demonstrate the integration of robots into CIM-Systems [1,2]. The critical path of this integration concerns the operational level of CIM-systems (fig. 1) and includes two closely interrelated fields of research: the realization of a computerized production planning system and of an off-line programming system.

At the beginning of the project a gross reference model of the overall system has been elaborated (fig. 2). As to be seen in fig. 2 the programming system has been subdivided into an implicit or automatic and an explicit or interactive programming system. Each of the three main components has access to a data base to models needed for simulation and programming as well as to the graphic and simulation system.

Based on this gross reference model of the overall system a functional specification of the mentioned subsystems has been elaborated. This project phase was followed by the realization of prototype systems which have been also successfully industrial applied.

In the following the concept for the execution of the project is presented. Furthermore a description of the subsystems is given and the most important results are mentioned.

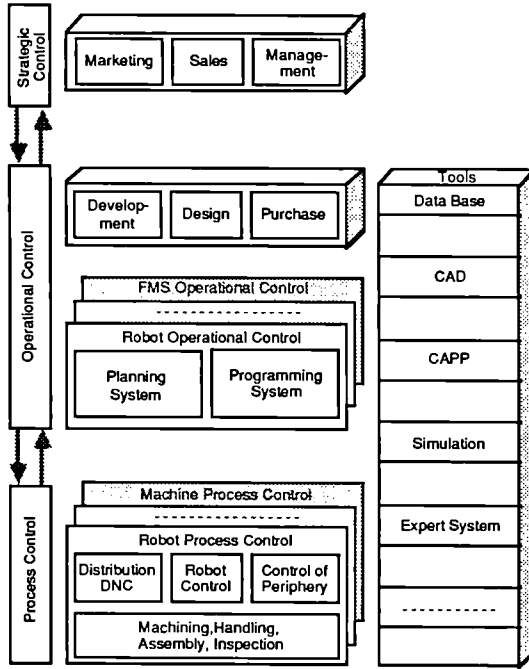


Figure 1
Reference Model for Robot Integration into CIM

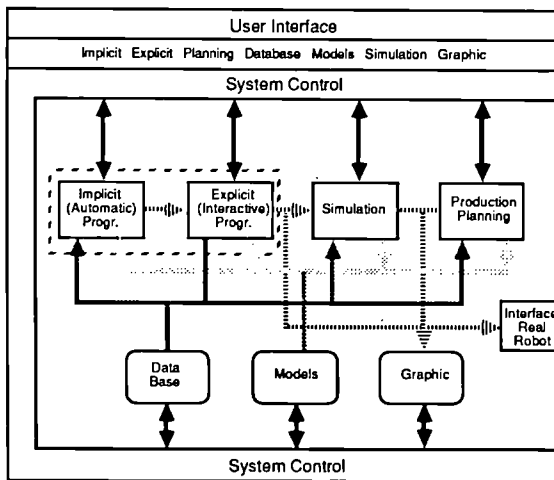


Figure 2
Reference Model of the Overall System

2. BASIC CONCEPT FOR THE PROJECT EXECUTION

In the following the main ideas for the execution of the project are pointed out. Within a pilot project the problem of the integration of robots into CIM-systems have been analyzed. As a result a list of design rules representing the definition of requirements has been elaborated [3]. This information has been used for the functional analysis of the conception and realization of robotized work cells. This functional approach during system specification led to a decomposition of the overall project task into the research areas computerized planning and off-line programming of robotized production systems (fig. 3)

In this context programming is to be seen not only as a formal generation of application programs. This is a task of minor importance which can be executed automatically. Programming is mainly a part of the production system planning process at a deeper level of detailing. Nevertheless those code generation and program transfer modules are specified and realized within the running projekt.

While in the systems planning phase which is executed before programming the robots task is defined in principle, e.g. move part from position A to position B in the programming phase a detailed determination is required. This means all parameters defining the trajectory and the technological process (e.g. path, tool orientations, velocity, welding time) have to be determined. Additionally optimization procedures for time or energy optimal trajectories can be applied.

A functional analysis of the programming task led to a separation into an implicit or automatic programming and an explicit or interactive programming component of the overall system.

As shown in fig. 3 the result of the functional analysis of the overall system is a decomposition into the three subsystems automatic programming, interactive programming and systems planning. For the further work on these subsystems three subgroups have been installed. After defining the requirements for each subsystem a functional analysis has been carried out leading to a functional specification. Based on these a detailed structure of each subsystem has been elaborated by each subgroup.

For the realization of the subsystems different approaches have been adopted for the following reason. For interactive programming and systems planning it seems to be possible to build up prototype systems which can partly realized with already existing software modules available by the different project partners. Therefore it was decided to build those systems using the method of fast prototyping. So far the first prototype systems for planning and programming have been realized and practically proved.

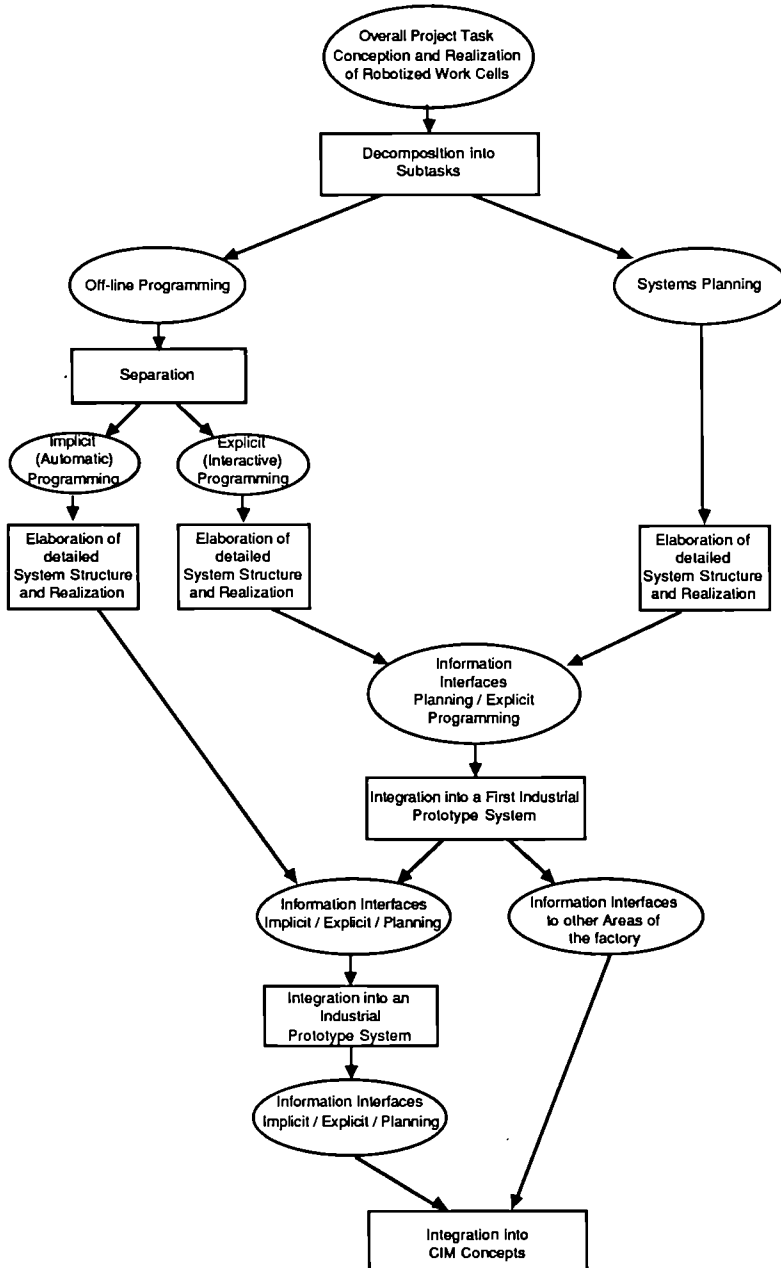


Figure 3
Concept of the Execution of the Project

Clearly these prototype systems could not provide all features planned but they demonstrated the whole functionality in principle and they are proved to be very efficient tools for the refinement of the functional specifications, especially for the definition of internal and external interfaces. They can also be used as test beds for software modules under development. Furthermore it turns out that the prototype systems could be applied successfully for different industrial projects. A short description of these prototype systems is given in chapter 3.

For the realization of the automatic or implicit programming system two concepts had been elaborated. The first one is a top-down approach, i.e. starting with realization after a detailed functional specification for the entire system and for each module and submodule. The other one is a bottom-up approach, i.e. realization of well specified modules which can also be used for interactive programming. An example for this is an automatic trajectory planning and optimization module. With interactive programming methods the system user is not able to define optimal trajectory neither related to time nor to energy. For that the robot's dynamic behavior has to be considered. The idea behind the bottom-up approach is a stepwise realization of the implicit programming system by automizing step by step programming functions which has to be fulfilled by the user during explicit programming.

So far the decomposition led to the three mentioned subsystems which could be more or less realized independent from each other. It is clear that for the necessary information exchange between those subsystems adequate interfaces have to be considered. But these interfaces have not to be defined in detail at this project stage only the type of information have to be determined.

This is not valid for the next project goal which is the realization of an industrial prototype system. This system will be built up by the integration of the subsystems interactive programming and systems planning. Therefore a clear definition of the information interfaces is required. This integrated system will then provide the whole functionality for the planning and programming of robotized work cells.

At the next step this system has to be integrated into a CIM environment. This means the information interfaces to other areas within a factory have to be defined. After that a pilot installation of the system in a company is planned.

A further step which can be done fairly exact in parallel is the detailed definition of information interfaces between the explicit programming system and the production systems planning at the one side and the implicit programming system at the other side.

The main information interfaces of the different subsystems are shown in the ISAC diagram of the overall system (fig. 4).

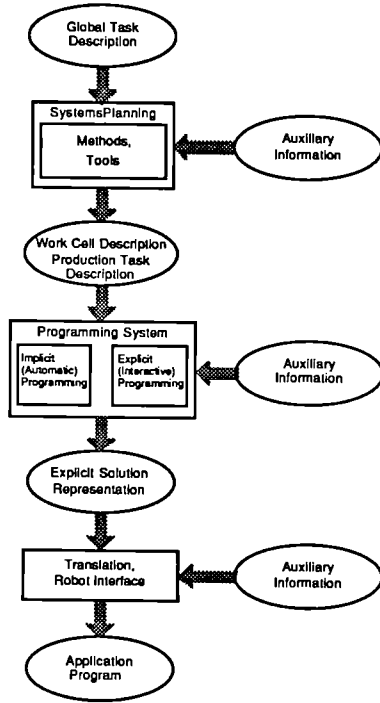


Figure 4
ISAC Diagram of the Overall System

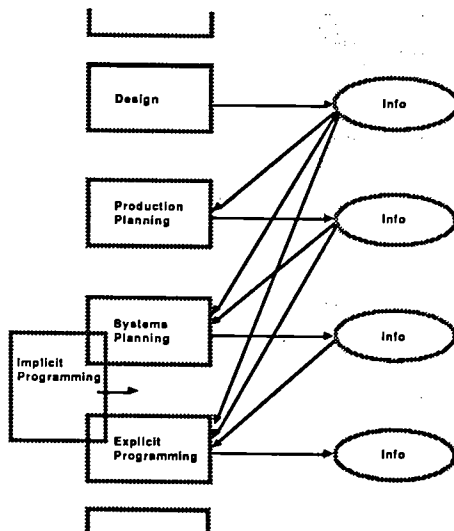


Figure 5
Considered Functional Subsystem for the
Integration into CIM-Environment

The most important information produced by the production systems planning necessary for programming, are descriptions of the work cell and of the production task. Using this information and additionally those from the world model the explicit solution representation can be generated. The explicit solution representation is a system internal information representation which contains at the end of the programming process all information necessary for the generation of explicit application programs. After their test of executability by means of simulation and test system they can be transferred to the real production system.

The above mentioned information integration aspect related to the information flow between the overall system under development and other information systems within the factory is shown in fig. 5. Also the main information sources are outlined. These are namely CAD-systems from the design area and data bases containing technology data and production facility data available in the production planning area. To reach the objective of an computer integrated manufacturing system the main problem to be solved is the integration of information produced by the different functional subsystems of the factory.

Applying the approach mentioned in the previous sections a result of the project work can also be producing guide-lines for information interface standards. This aspect is also important for the combination of different tools like CAD-systems and data bases.

3. DESCRIPTION OF SUBSYSTEMS AND MAIN RESULTS

In this chapter a description of the three subsystems implicit (automatic) programming, explicit (interactive) programming and systems planning is given. Additionally the main results are pointed out.

3.1 Implicit Programming System

The implicit (automatic) programming system will allow to specify the robot task through a task level specification instead of a robot level specification. That means the task is described as a sequence of changes in a model of the environment. Thus an action is specified by its desired effects on objects of the model, rather than specifying the motions necessary to achieve them. The system then plans these actions and generates a robot level specification of the task.

The structure of the implicit programming system is shown in fig.6. The operator specifies via an user interface the production task. It contains different modules which are covered mainly by the subgroups systems planning and explicit programming. Also tools are integrated for the textual and graphical dialogue with the user for information aquisition. The information (problem decription) is used as the input for the planning system.

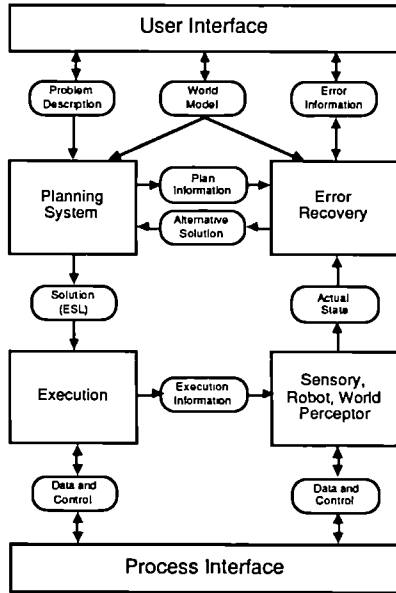


Figure 6
Reference Model of the Implicit Programming System

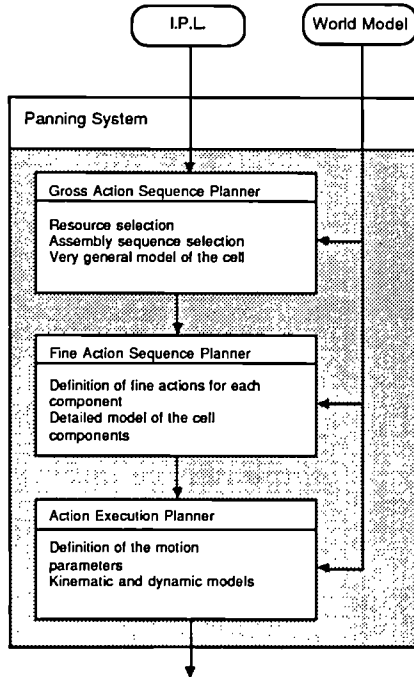


Figure 7
Gross and Fine Action Sequence Planner

Using this problem description and information from the world model the planning system as a central part generates a plan of discrete operations which solve the specified problem.

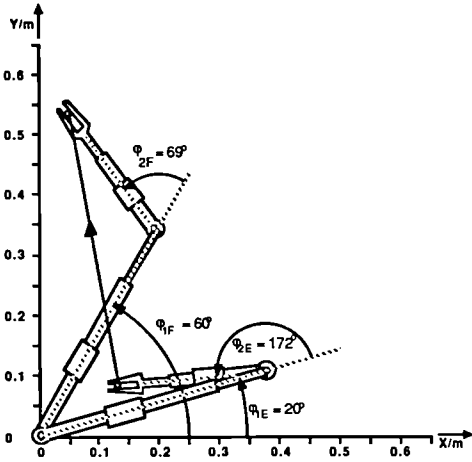
In principle, this is executed in two steps. In the first step the plan to be generated is structured, i.e. the task execution sequence is defined. Thereby is distinguished between the gross action sequence planner responsible for the interaction of production system components and the fine action sequence planner [4]. The second one creates out of the global task sequence, delivered from the gross action sequence planner, the task execution sequence for each system component.

In the second step the produced information from the action sequence planner and from the world model is used by the action execution planner which is responsible for the planning of the operation details. The modul consists of a set of subplanners specialized on the different planning problems. The output information is dependent on motion specification and on the specific subplanner appropriate for a particular class of planning problems, e. g. grasp planning or robot motion planning. As a general term for the output information explicit solution representation has been chosen.

Especially for the robot this means all information necessary for application program generation are included in the explicit solution representation which is one of the interfaces to the explicit programming system.

The module execution which is mainly a module of explicit programming is responsible for producing all necessary information for controlling the robot cell via the process interface. The sensory robot and world perceptor derives the actual state of the world and the robot during the execution by interpretation of sensor information. Several sensor devices monitoring the programm execution in the physical world. They are connected to the submodules "integrator" and "interrupt manager". The function of the integrator is to link different sensors together, integrating the different types of information they provide into a form suitable for further processing. The interrupt manager is an expert system, that can interpret data provided by the integrators, detect emergency situations and correctly dispatch sensory information to other modules of the system.

The output information is delivered to the error recovery module. It compares the actual state with the intended state, which is contained in the generated plan. On the base of this knowledge it should be possible to rebuilt the generated plan. The main component in the error recovery module is the module "error analyser and planner". This module detects an error situation by comparing the actual state in the physical world with the state which was intended by the planning system. The intended state is contained in the plan information which represents the structure and the elements of the generated plan. The actual state in the robot cell is derived by the interpretation of sensor signals which are monitored during the program execution. This work is mainly



Given Path and Kinematic Konfiguration

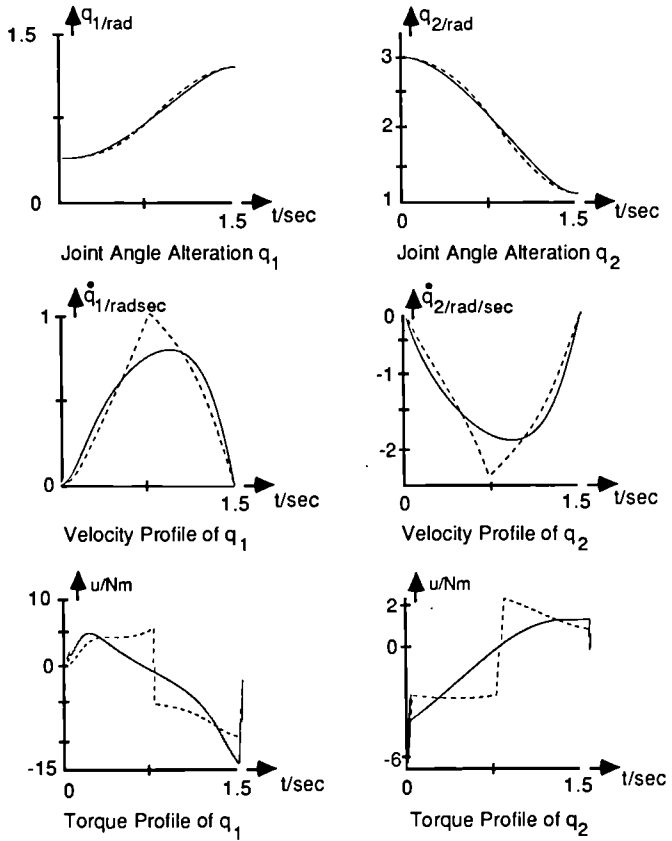


Figure 8
 Profile for the Optimized and Non-Optimized Case for
 a Given Path and Kinematic Konfiguration

done by the sensory robot and world perceptor. If the error is detected and recognized a description of the error a local planning facility is provided which can provide an alternative solution for the special problem.

The long term objective of implicit programming is the creation of a system for automatic robot program generation. First results in this area have been demonstrated in April 1987 at a review meeting in Milano:

- The action sequence planner (fig. 7) which has been applied to the assembly of the Cranfield Benchmark. The actual second version has been implemented in OPS 83. For the non inferential part PASCAL has been used.
- An algorithm for energy optimal trajectory planning for CP motion, which is a module of robot execution planning. For the demonstration of the developed algorithmen which is based on Bellman's Principle a kinematic with two rotary joints was selected (fig. 8).
- The skeleton extractor which is a module of error recovery (fig.9). For the demonstration "peg in hole" operations were chosen. The classes of errors that can be detected are: parts defective, parts misaligned, parts not correct positioned, parts fall from tool and trajectory is not collision free. The realized software has been implemented in OPS 5 (inferential part) and in PASCAL.

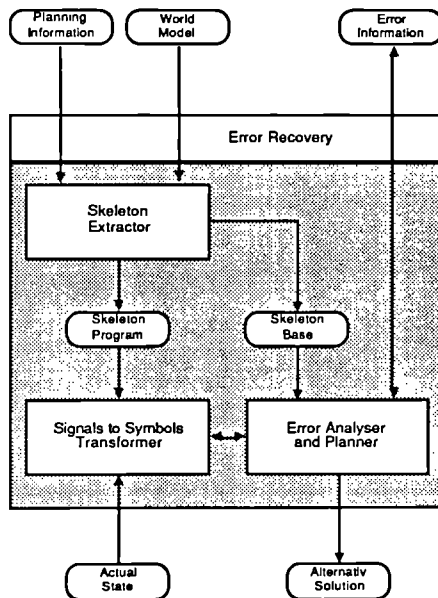


Figure 9
Skeleton Extractor

3.2 Explicit Programming System

In parallel to the automatic robot programming activities, for short term introduction of off-line programming techniques into industry an explicit programming system is under development. The objective is an interactive procedure for robot program generation. During the process the user has access to company internal information systems and is supported by special system functions to increase the efficiency of the programming process. To guarantee the executability of robot programs as well as the definition of cycle times a simulation system is an integral part. Prototype systems have been developed and industrial applied.

The principal structure of this system is shown in fig.10. The figure gives an overview of the necessary interfaces to the system user, to data bases and models and to the real robot. Further the integration of debugging and test functions (simulation and graphic system) is outlined. IRDATA is used as an interface as well as for the simulation system and for the real robot.

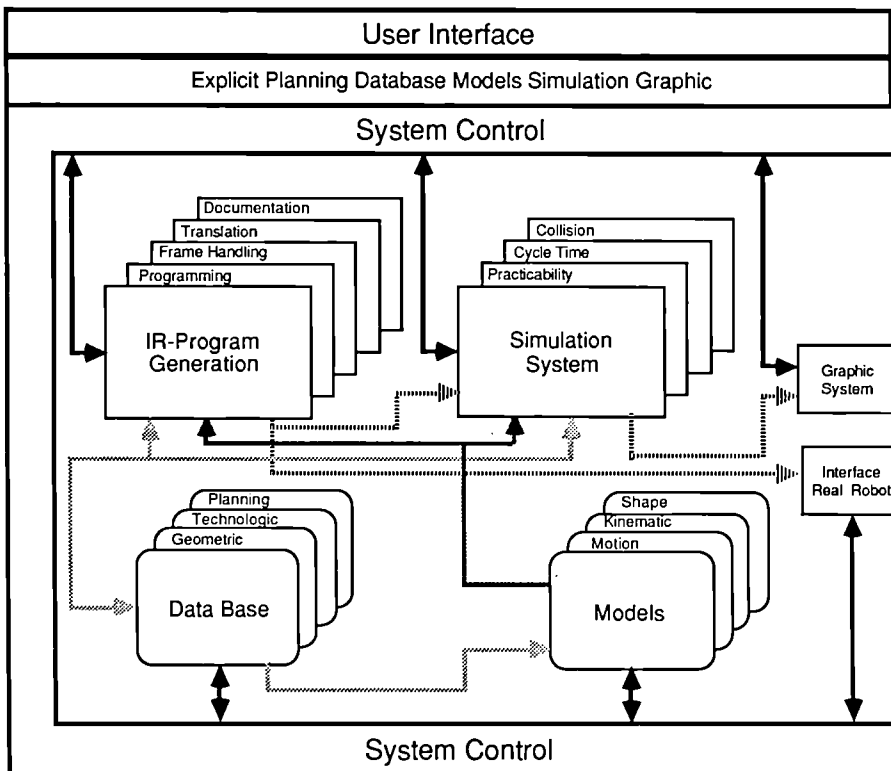


Figure 10
Reference Model of the Explicit Programming System

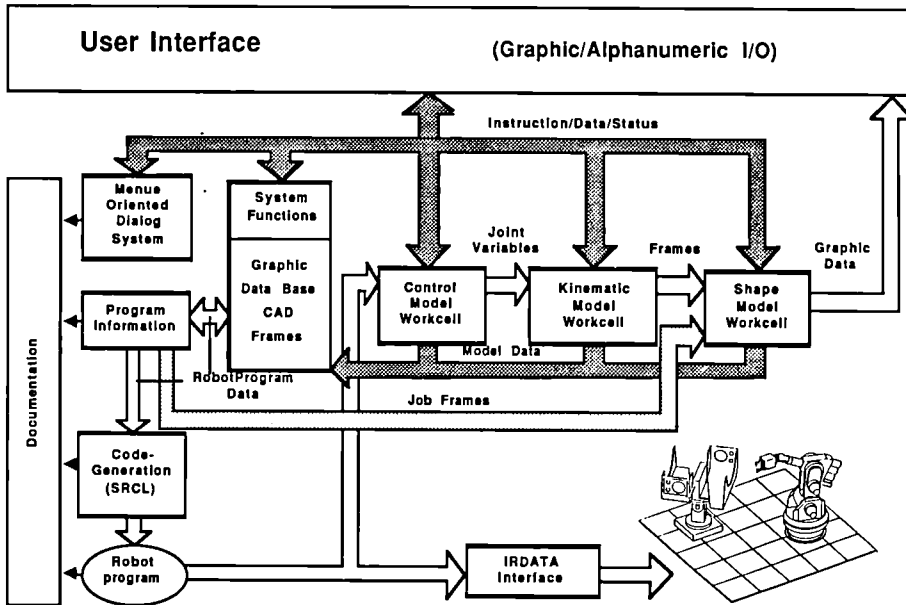


Figure 11
Functional Structure of a Prototype System

The structure of a realized first prototype is shown in fig. 11. The user communicates via a dialog oriented interface with the system. The interface has graphic interactive and alphanumeric features. Additionally the user is enabled to activate system functions for e.g. data base access or the manipulation of frames. An important part of IR-application programs are geometric data describing the position and orientation of the end-effector system. These data are partly available from or can be derived out of the CAD-system. For the definition of the geometric IR-application program data suitable system functions for the access to different CAD-systems (ROMULUS, EUCLID and COMPAC/APS) have been realized.

An important aspect concerning the requirements for task description, application program creation and simulation is related to the determination of frames and frame relations (frame concept) [5]. This frame concept contains the frame relations describing the cell components related to the cell reference system and the subcomponents of a component related to the component reference system.

For the frame concept homogenous coordinates are used. All frame relations can be calculated by uniform matrix operations. The mathematical expressions are 4x4 matrices which describe the configuration of each frame.

The technological IR-application program data (e.g. velocity) is requested by the system and if required defined by the user. As a result a data structure called program information is created which contains all information describing the robots task independent from a specific robot control and a specific robot programming language. Using the program information the determined positions and orientations of the tool and sequences of job frames can be tested independent of a specific robot control. Furthermore visual collision checks of the workpiece/tool and cell arrangement can be realized.

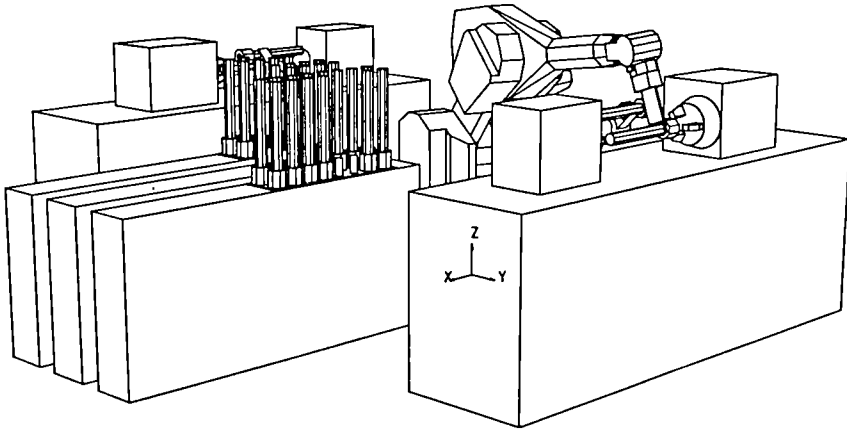
Out of the program information an explicit robot program language is created by the module code generation. Due to the demand that off-line created robot programs are practicable to a maximum extend on the shop floor a simulation and test is required.

Therefore the robot programs are transferred to the robot control model which interprets the program and calculates the robot joint variables. Out of these values within the kinematic model frames are calculated which describe position and orientation of each component related to the kinematic chain of the robot. For the calculation of the configuration of other components the respective kinematic model of the component is required. The kinematic model of the work cell transforms all local defined frames into a common system, e.g. the cell reference system. This procedure is necessary to enable the generation of all required frame relations between cell components.

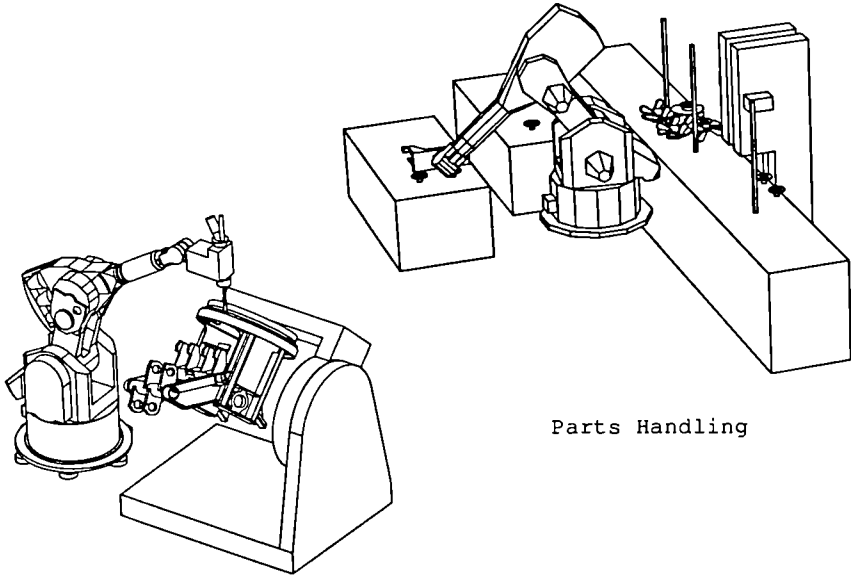
By these frames each shape model is configured in the work cell and visualized by the graphic system. After the translation of robot programs into the control internal code or IRDATA they can be transferred to the real robot system via the DNC-interface.

As mentioned above first testbeds of off-line programming systems have been installed by different project partners and also applied for industrial projects. The following systems were demonstrated at a review meeting in October 1986 at Berlin:

- A programming and simulation system which has been used for different industrial projects (fig. 12). To demonstrate the range of applications three typical examples are seam welding of a robot base, handling of parts of a driving collar and loading and deloading of two turning machines. The system is implemented in FORTRAN and PASCAL. The CAD-System ROMULUS is used.
- The PREDESIGN system which has been applied for the insertion of rings into holes of an aluminium block (fig. 13). The system can be used for the specification as well as for cell modelling and programming. The system is implemented in PASCAL. The CAD-system EUCLID is used.
- The robot simulation system (ROSI). The system's main goal is the simulation of an automatized workcell, in order to allow planning and programming of real cells in production lines considering the integration of robot. Some examples demonstrating the features of ROSI are shown in fig. 14. The system is implemented in PASCAL. The CAD-systems ROMULUS and



Loading and Deloading of two Turning Machines



Parts Handling

Seam Welding of a Robot Base

Figure 12
Industrial Application Examples

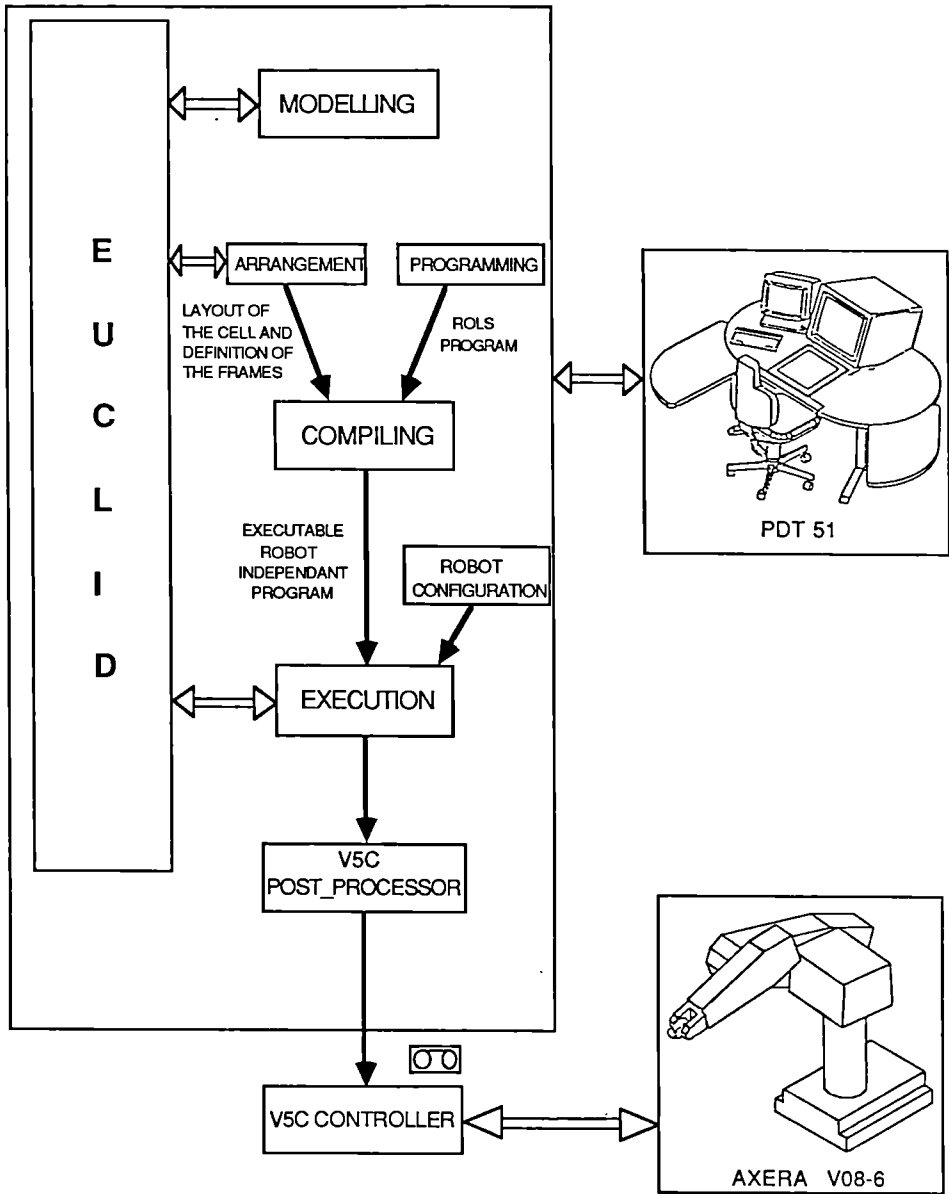


Figure 13
 Prototype of Predesign System

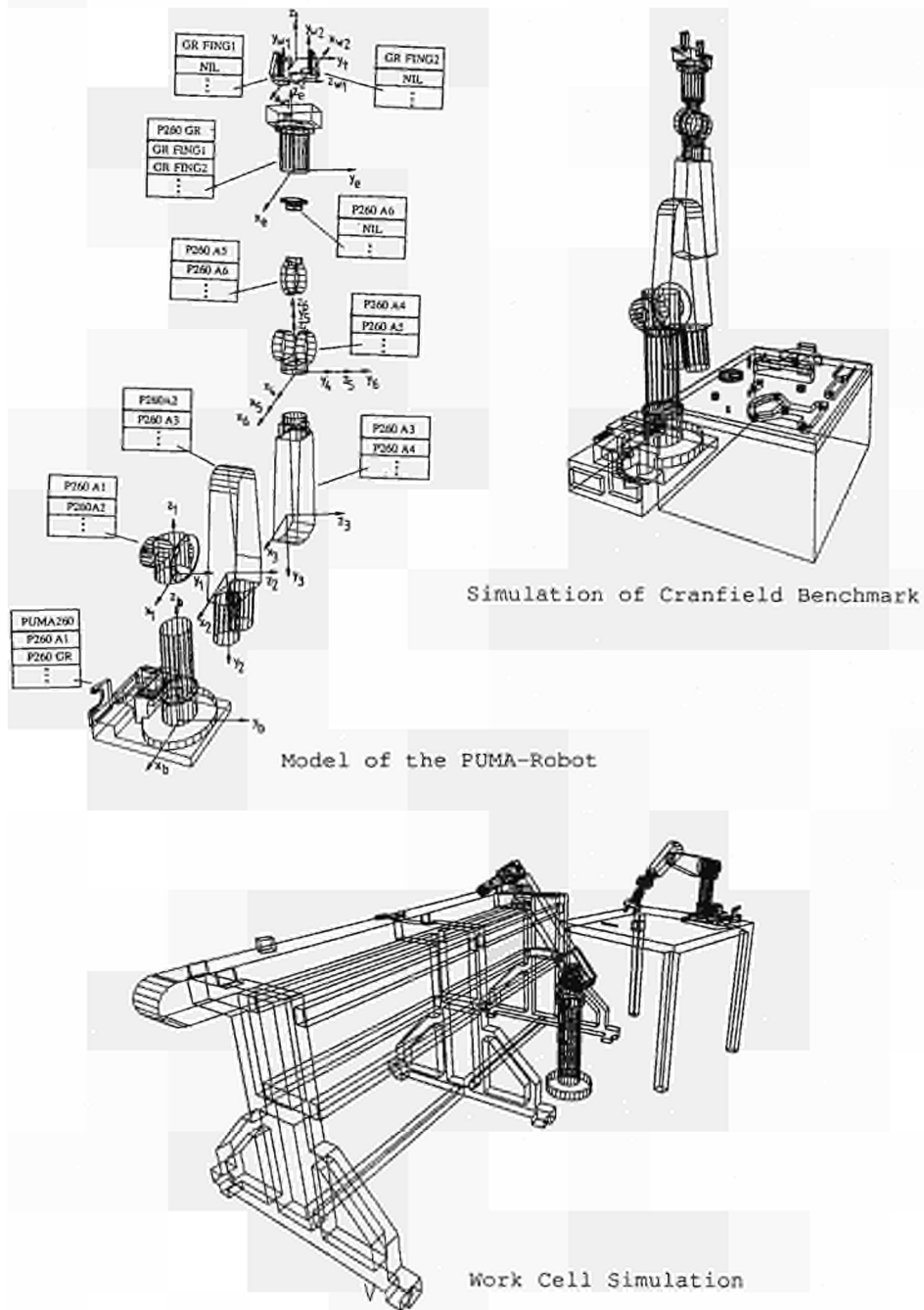
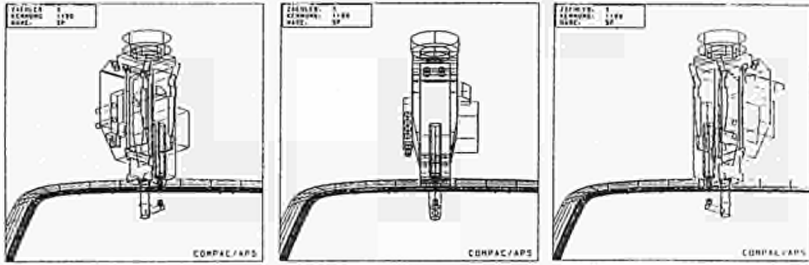
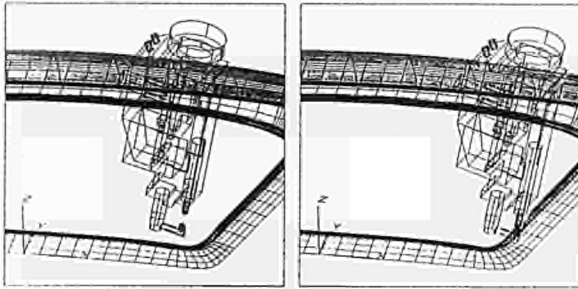


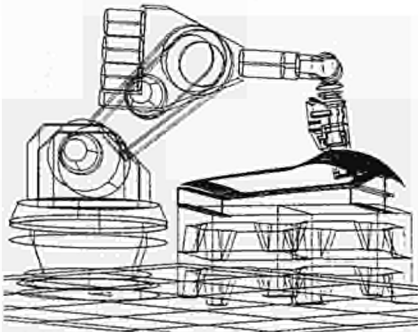
Figure 14
Robot Simulation System (ROSI)



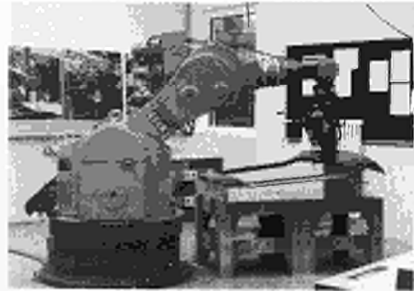
Definition of Gun Orientation and Intermediate Points



IR-Independent Simulation and Test



IR-Dependent Simulation and Test



Application Program Execution at the Real Robot

Figure 15
Application Program Generation, Test and Execution Procedure

EUCLID are used.

- An off-line programming and simulation system which has been applied for spot welding (fig. 15). The full range of functionality of the developed system from off-line program creation, simulation and test to the execution of the application program at the real robot in automatic mode was demonstrated. The system is implemented in FORTRAN. The CAD-system COMPAC/APS is used.

As a summary of the experience by applying the interactive programming systems it turns out that the systems are not only useful for the off-line generation and test of robot application programs. Several functions and modules are also useful to support the design and the production planning process [6]. An example for the first area is the determination of welding spots during workpiece design. An example for the latter area is the selection of the spot welding gun. Additionally the developed simulation systems can be seen as important tools for systems planning. Furthermore it turns out that the developed systems are first steps towards an integrated manufacturing system whereby a complete information flow from the design area over the production planning and the application programming down to the production area is realized.

3.3 Production Systems Planning

The demand to react to variable situations of the market forces manufactures to design flexible production equipment. Due to a decrease of product life terms and increase of product complexity, the planning requirements and influences will become more important within the future industrial manufacturing. These are characterized by more frequent planning procedures, planning tasks under short time constraints, increase of planning complexity and efforts.

Particularly in the area of assembly, where all partstreams and product informations merge, a most effective planning system is required. For the development of a planning system the following objectives are essential :

- improvement of information flow,
- enhancement of planning transparency,
- high level of planning quality and
- reduction of routine activities.

A planning system consists of two major components. One is a methodology, which guides the planner from the planning task to the planning solution. The other important part are suitable tools for the execution of the planning process (fig. 16).

Concerning the first area a planning methodology has been elaborated consisting of seven planning steps [7] which are briefly explained in the following.

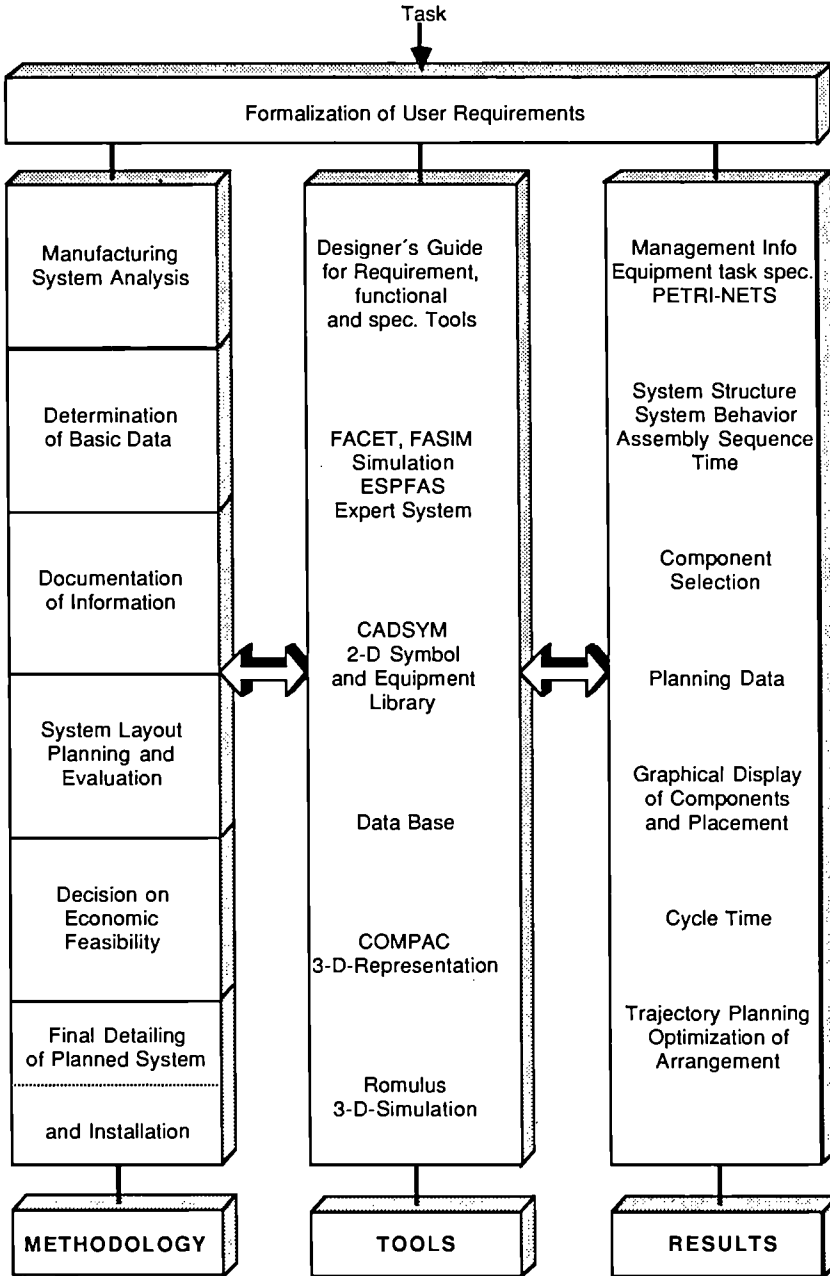


Figure 16
Reference Model of System Planning

1 Manufacturing System Analysis

A detailed analysis of the actual state delivers the basic data for the further planning steps. These data are related to the manufacturing process, product, organizational and production control and to the human operator.

2 Determination of Basis Data

The compilation of basic data has the function to enable a precise formulation of the planning task. In the transition from an existing system to a planned system additional information is required.

3 Documentation of Information

After defining and structuring the task, the process of detailed data preparation follows. The result of this descriptive information processing is the compilation of all demands and wishes in a list of requirements.

4 System Layout Planning and Evaluation

Based on the previous data in this step a concept of production equipment and their geometrical relations will be generated and evaluated. The planning procedure is further divided into the four categories: generation of solutions, solution detailing, solution optimization and evaluation.

5 Decision on Economic Feasibility

After the technological prerequisites have been worked out in the previous steps, relevant aspects of economy are considered. These considerations are meant to serve as a base for deciding whether to realize a suggested solution or not. The goal of this entire step is to get a rough estimation of economic parameters, without spending additional time and costs by further detailing of the planned system. Only if the results are in a suitable economical range, the next step will be processed.

6 Final Detailing of Planning System

The aims of this step are to work out all required data needed for pilot production set up, for equipment on the market available and devices, which will be designed and manufactured in house.

7 Planning of Systems Installation

In order to minimize manufacturing interruptions during the introduction of new system installations, a strategy has to be developed which is based on organizational-, economical-, technical- and human-aspects.

According to this planning methodology suitable tools have been analysed and are under development. Based on a 2D- and 3D - CAD - system as well as additional software tools for specific planning tasks complex assembly systems can be realized. Such tools under development are FASIM, FASET and ESPFAS. FASIM is a data driven generalized simulation model for flexible assembly systems design and evaluation which incorporates the use of another generalized simulation model, FACET.

FACET is a generalized model of a single cell, i.e. certain features are offered and the user chooses from these to suit his needs. When this cell has been specified the user runs the model which generates a distribution of cycle times for the cell. This distribution can then be passed down to FASIM which models the cell as a black box.

FASIM, like FASET, is a data driven generalized simulation model but operates at the system level rather than the cell level. It simulates the flow of assemblies, resources and information through the system. Its data input covers the hardware and control aspects of system design. The performance output consists of throughput time, cycle time, utilizations, queue statistics etc.

ESPFAS is a pre-prototype expert system for assembly planning in robot based flexible assembly systems.

The production planning system is used for selection of suitable assembly devices as well as generation of alternative layouts. Furthermore it supports statical and dynamical simulation and defines geometrical parameters, cycle times or investment costs. For feedback purposes with regard to computer-aided robot planning system a set-up flexible assembly cell was developed and installed.

The planning of complex and robot guided systems requires expert knowledge which can be classified into the domains shown in fig. 17. While the previous project work was mainly focussed on methodological approaches, now the investigation of the relationship between planned virtual assembly cells and installed equipment was started. With the aid of the realized assembly cell it is possible to check, test and improve the planning knowledge in a wide area of assembly tasks.

One of the goals for the automated assembly cell installed have been to perform assembly tasks of different products simultaneously in a flexible sequence. Firm requirements are well defined set-up conditions of robots and peripheral equipment.

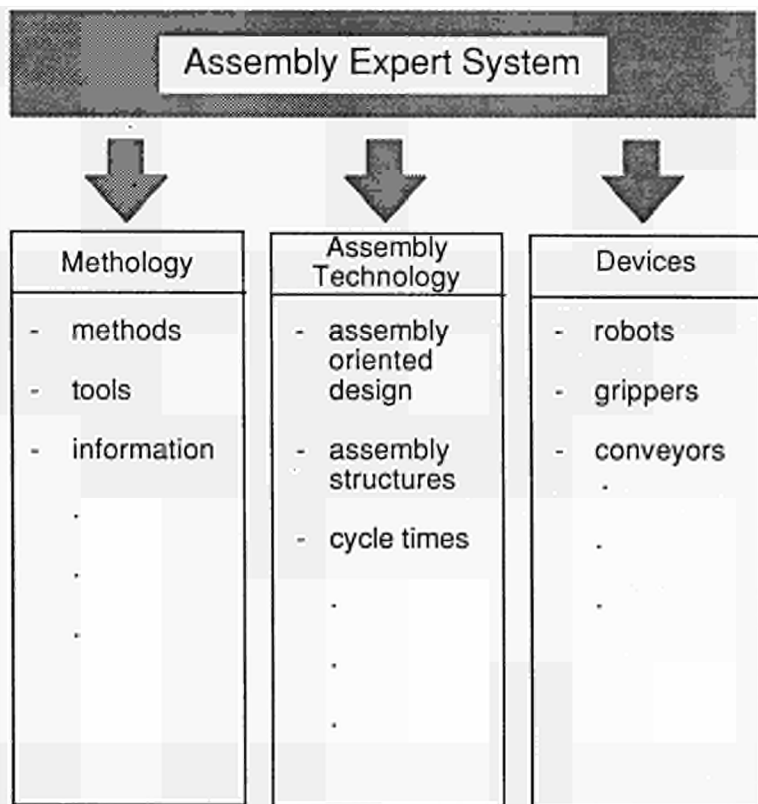


Figure 17
Required Knowledge for Assembly Planning

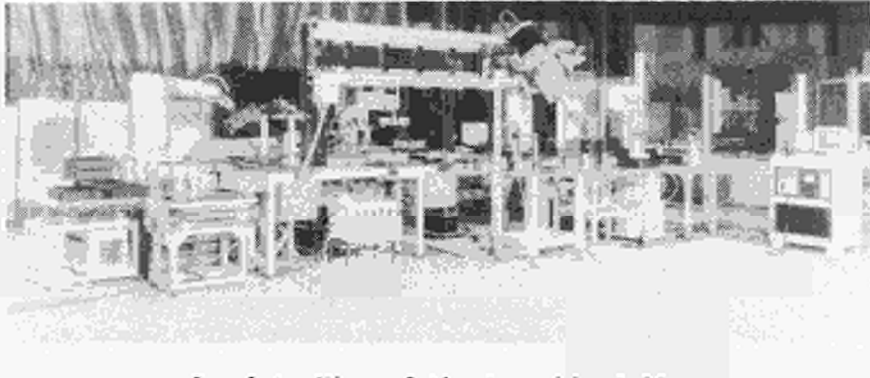
Basic demands for the automated set-up process are reproducible positions at the robot stations. This has been achieved by standardization of pallettes, part presentations, the fixtures for the carrier system, the gripper exchange system as well as part and tool-carrier which can be indexed and coded.

Two kinds of working operations have been defined for the execution of different assembly tasks. These are the set-up process and the automatic operation.

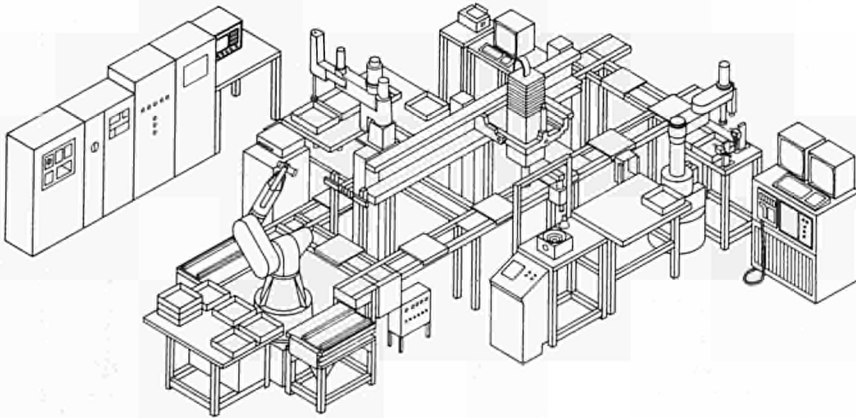
During the automatic operation it is possible to assemble different product variants.

First results in this area were demonstrated in October 1986 in Berlin:

- A realized assembly cell consisting of four different robots and a flexible material flow system (fig. 18). As a first prototype task the assembly of different types of matrix printer bodies was chosen.
- FACET and FASIM and the preprototype version of ESPFAS for robot based flexible assembly systems (fig. 19). As demonstration examples the assembly of a telephone and an electric plug have been chosen. For implementation OPS 5, SLAM II and FORTRAN have been used.

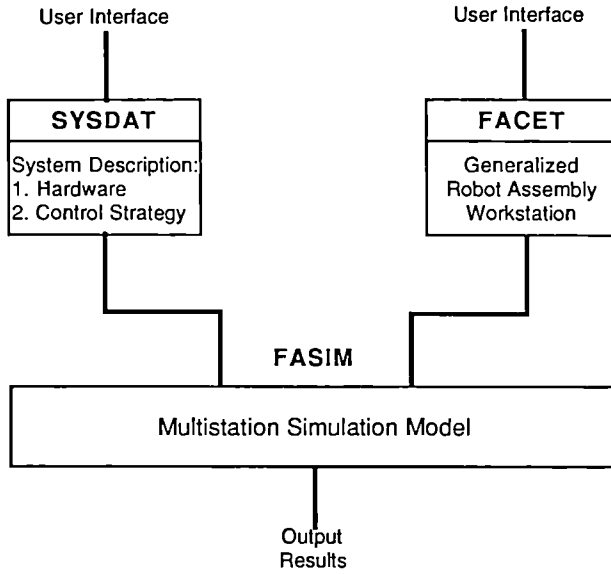


Complete View of the Assembly Cell

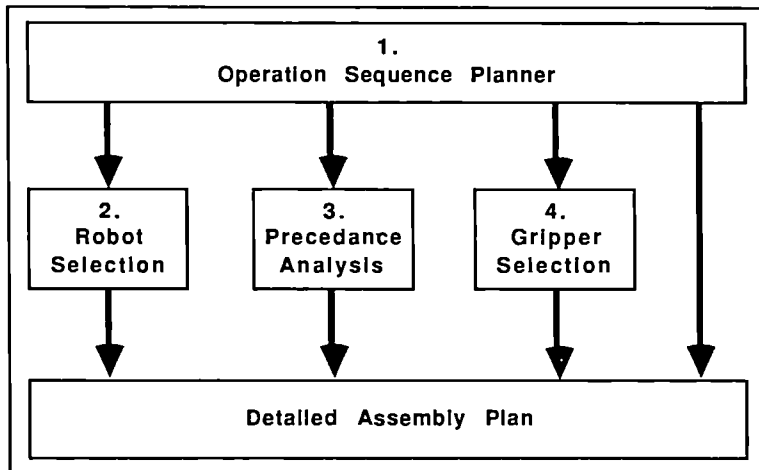


Configuration of the Assembly Cell

Figure 18
Realized Assembly Cell



Operation of FASIM



Flow of Data in the Assembly Planning System (ESPFAS)

Figure 19
Assembly Planning Prototype System

4. CONCLUSION

In the first project phase the overall system had been decomposed into the three subsystems systems planning, explicit and implicit programming. For the first and the second one prototype systems have been developed and successfully applied for industrial tasks. These prototype systems are also very efficient tools for the refinement of the functional specification. Furthermore they are used as test beds for different software modules under development.

Due to the fact that implicit programming is more research oriented two concepts have been elaborated. The first one is a top-down approach starting with a detailed functional specification. Additionally a bottom-up approach has been applied to realize well specified submodules. This approach has been chosen to automatize programming subtasks which had to be otherwise fulfilled by the user.

The information integration aspect is the main problem for the realization of an industrial prototype system combining programming and system planning. Also the integration into the overall CIM environment has to be considered. These topics are parts of the actual research work.

REFERENCES:

- [1] G.Spur et al Integration of Industrial Robots into CIM-Systems SITEF Toulouse, Oct. 24th, 1985
- [2] U.Rembold,
M.Vojnovic Operational Control for Robot System Integration into CIM IEEE 1986, International Conf. on Robotics and Automation, April 7th-10th,1986, San Francisco
- [3] G.Spur et al Design Rules for the Integration of Industrial Robots into CIM-Systems (Final Report of Project Number 75)
- [4] B.Frommherz
K.Hoermann Ein Konzept für ein Roboter Aktionsplanungssystem, 16. Jahrestagung der Gesellschaft für Informatik, Berlin 1986
- [5] G.Duelen,
U.Kirchhoff
R.Bernhardt
G.Schreck Algorithmic Representation of Work Cells and Task Description for Off-line Programming, International Conference on Intelligent Manufacturing Systems, Budapest, 1986
- [6] G.Duelen,
R.Bernhardt,
H.Linnemann Informationsarchitektur in datengetriebenen Fabriken, Produktionstechnisches Kolloquium, Berlin 1986
- [7] G.Spur,
I.Furgac,
J.Browne,
A.Deutschländer,

ACKNOWLEDGEMENT

The work is supported financially by the European Community within the ESPRIT program (ESPRIT project 623). The following researcher and institutions are involved:

- G. Spur (Project Leader), U.Kirchhoff (Project Manager), W.Felsing, R.Bernhardt, F.Severin, A.Deutschländer, M.Dlabka, G.Schreck, V.Katschinski, Fraunhofer-Institut für Produktionsanlagen und Konstruktionstechnik (IPK), Berlin, Federal Republic of Germany (Prime Contractor).
- S.Milani, F.Rodighiero, FIAR SpA, Divisione Spazio e Automazione, Milano, Italy (Contractor).
- H. Wörn, G.Stark, D.Schwendinger, W.Miosga, G.Schratzstaller, E.Postenrieder, KUKA Schweißanlagen und Robotertechnik GmbH, Augsburg,

- Federal Republic of Germany (Contractor).
- D.Vojnovic, B.Chenu, J.P.Schiltz, P.Auger, F.Lemoine, Renault Automation, Paris, France (Contractor).
 - J. Browne, S.Wadhwa, C.Tierney, R.Bowden, P.O'Gorman, University College Galway, Department of Industrial Engineering, Galway, Republic of Ireland (Contractor).
 - U.Rembold, R.Dillmann, B.Frommherz, T.Soetadij, U.Negretto, M.Huck, K.Hörmann, Lehrstuhl für Prozeßrechentechnik, Universität Karlsruhe, Federal Republic of Germany (Contractor).
 - A.Steiger, L.Camarinha, J.Afonso, Universidade Nova de Lisboa, Portugal (Contractor).
 - E.A.Puente, A.Jiminez, F.Sastron, Universidad Politecnica de Madrid, Spain (Contractor).
 - E.Pagello, P.Bison, LADSEB -CNR, Consiglio nazionale delle Ricerche Istituto per Ricerche di Dinamica dei Sistemi, Padova, Italy (Subcontractor).
 - M.Somalvico, R.Cassinis, G.Gini, Politecnico di Milano, Milano, Italy (Subcontractor).
 - W.Jakob, Gesellschaft für Prozeßsteuerung- und Informationssysteme mbH (PSI), Berlin, Federal Republic of Germany (Subcontractor).
 - L.O.Hertzberger, F.Tuynman, G.R.Meijer, University of Amsterdam, Netherlands (Subcontractor).
 - Baracco, A.LeRex, M.Haihouj, SERAM, Societe d'Etudes et de Recherches de l'Ecole Nationale Superieure d'Arts et Metiers, Paris, France (Subcontractor).
 - R.Soenen, Y.Salez, Université de Valenciennes, France (Subcontractor).

Project No. 909

C-BAT : A METHODOLOGY AND SOFTWARE TOOLKIT TO HELP SMALL-TO MEDIUM SIZED FIRMS TO ASSESS CIM.

Dr. M. Koriba AMTRI, Hulley Road, Macclesfield, SK10 2NE, UK.

P. K. EYRES AMTRI, Hulley Road, Macclesfield, SK10 2NE, UK.

Project 909 (C-BAT) was started in 1986 to devise a methodology and supporting software 'toolkit' for helping Small to Medium-sized Manufacturing Enterprises (SMEs) in assessing the costs and benefits of Computer-Integrated Manufacturing (CIM).

Following an introduction which sets the scene for the project, this paper presents the following three aspects:

- the major findings of a survey of companies with experience in assessing CIM.
- the results of the Project as it approaches its mid-point
- a look ahead to the emerging Cost-Benefit Analysis Toolkit (C-BAT)

1. INTRODUCTION

1.1 Background to Computer-Integrated Manufacturing

Computer-Integrated Manufacturing (CIM) can play a major role in improving the competitiveness of European Manufacturing Industry: e.g maintain competitive manufacturing costs in the rapid changes in design and product mix, and increase production volumes while holding small inventories and minimising work-in-progress.

However, there are still problems related to the adoption of CIM. These problems can be grouped under three heading: Technology, Management and Economics.

Technology: CIM cannot happen overnight. Many Computer-Aided Technologies require to interact with each other in a fully integrated manner, and according to some architecture. This latter must show the inter-relationship between these Computer-Aided Technologies, and the timely flow of relevant information between those activities of a company which are computer-assisted. Work in this area is being covered by other ESPRIT projects (CIM-OSA and CNMA)

Management: The commitment of company staff is an essential pre-requisite to effective adoption of CIM. Rationalisation of the company's activities and their subsequent integration must be reflected in the management structure and practices in many cases.

Economics: Conventional accounting techniques cannot determine all of the costs and benefits associated with the introduction of CIM. This is because they are not sufficiently developed to take into account less tangible parameters such as market-share, better product quality and reduced leadtime. The benefits, in fact, arise through greater flexibility and responsiveness, rather than from productivity improvements only..

The majority of Project 909's resources are devoted to tackling the management and economic problem-areas.

1.2 CIM in Small to Medium Enterprises (SMEs)

Large organisations have been able to implement and benefit most from CIM because:

- they are most likely to possess the required resources to address the above three problem-areas.
- they are more likely to be involved in a full range of activities; from marketing through sales, design/development and manufacturing, to distribution. They are in the position to integrate these activities, and, hence, improve their competitiveness.

It is not immediately apparent in the case of SMEs, how these same principles may be applied, because SMEs tend to be involved in a limited range of activities. Thus, integration of their activities may have to be extended to their partners, sub-contractors, suppliers, and so on, which is a much more complex task than integrating a company's intra-activities. Furthermore, many SMEs believe that the economics and scale of CIM are beyond their resources.

SMEs form a vital part of the European manufacturing industry and it is for this reason that the C-BAT methodology and toolkit are targetted for use principally by SMEs.

2. THE SURVEY

2.1 Introduction to Survey

A survey of companies with experience in CIM was carried out to gather some of their experiences in analysing investments in CIM. Within partners' member states, 66 organisations were visited by staff from the Project partners and a 12-page questionnaire was filled in for each company or organisation.

The answers to the question were analysed by the German partner (BIBA) using a computer-based technique which identifies similarities in the sets of data from the organisations.

2.2 Results of Survey

The majority of the results of the survey are presented in the Tables 1 to 9 at the end of the paper. Inevitably, not all questions were answered by the participants, so the values in the Tables represent only those who gave valid answers to the specific questions. Here are the major points, in brief, from the survey and the Tables:

- Nearly 70% of the participating organisations (the majority in the metal-working industry) employed less than 500 people (the EC definition of SMEs), although nearly 80% have annual turnovers in excess of 5 000 000 ECU.
- Over 60% were mainly-involved in 'batch' manufacture (Table 1).
- Over 70% claimed to 'Work to order' (Table 2).
- 70% used computers for administration, more than half of which linked their applications using a database (Table 3).
- More than 50% used computers for manufacturing management activities, and just less than half of them used a database to link applications. Manufacturing management was identified as the most likely growth area for the use of computers and linked applications (Table 3).
- In technical applications, CNC/DNC, CAD, CAM (NC part-programming), CAD/CAM and Production monitoring configurations were most popular (Table 4).
- Over 60% of the companies spent less than 500 000 ECU on their CIM configurations (Table 5).
- Over 60% stated that a corporate strategic plan controlled (or would control) their plans for implementing CIM concepts; nearly twice the number who indicated financial constraints as their major influences (Table 6). 36% claimed that they assessed the benefits of their investment for the whole organisation.
- The four most important long-term, strategic reasons for adopting CIM were:
 - maintain/reduce manufacturing costs
 - maintain market share
 - integrate information flow
 - improve product quality/reliability

- Very few were able to identify and quantify their predicted and attained values for the operating benefits of their systems: those who predicted large improvements were, in general, disappointed with their actual achievements. (Table 7 Percentage values based on very small numbers of valid responses)
- The major problems with investing in CIM were (Table 8):
 - difficulty to evaluate the economics
 - lack of tools available in the management area
 - lack of understanding of systems in the technological area
- The use of various conventional accounting techniques from (Table 9) could be the reason for the prediction of large productivity improvements that were subsequently not achieved from strategic decisions.

These latter findings confirmed the need for the C-BAT and identified the need for a methodology to be incorporated into the final concepts.

2.3 Survey Analysis: Four major company types

The analyses of all the participating organisations throughout the European Community identified four major types or groupings of manufacturing organisations:

- 'high-tech' organisations, usually employing large numbers of people and having high sales revenues

- large organisations, usually serving mature or stable markets, that require good manufacturing co-ordination to remain competitive

- small/medium-sized organisations with no specific production characteristics

- small organisations who concentrate upon 'jobbing' or small-batch products.

The 'high-tech' companies, especially the small ones, claimed that their **justification was not based solely upon traditional accounting methods**, but on:

- 'sound' technical judgements
- strong management commitment
- a 'team' approach to implementation
- a willingness to learn new methods (by all the staff concerned)

Within the large and medium-sized companies, technical staff used spreadsheet or self-written programs to help to cost-justify proposals against traditional accounting criteria.

There was little, or in most cases no, involvement of other departments (e.g. finance, marketing) in discussing potential benefits. However, there was an almost universally-acknowledged need for projects of high technical merit to be assessed against wider based criteria in order to sustain the enthusiasm of technical staff and avoid the, often false, emphasis upon reducing direct labour costs.

Many small or medium-sized organisations appeared to rely upon a single, responsible, senior member of staff to specify and implement the technology successfully - the so-called enthusiastic champion.

2.4 Conclusion of Survey

There was one major conclusion:

Those who have used a long-term plan (or strategy) have made best use of the current computer-based technologies and are in a good position to take advantage of developments of these technologies, when these are relevant to their business.

This is particularly apparent in:

the management's commitment and understanding of what is required of the technologies within the operations of the business

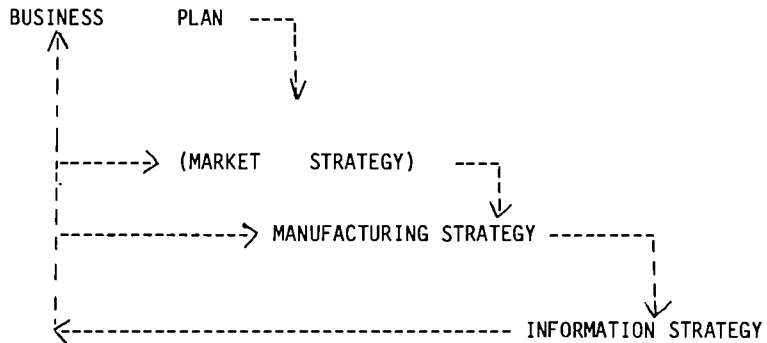
the education and training aspects needed to obtain the required performance of the new methods

A supplementary conclusion, allied to the major one above, is that, when short-term criteria are applied to CIM investments there is a temptation to exaggerate benefits, especially savings. This can lead to early disappointment and loss of confidence when these overstated benefits do not materialise and could be especially damaging if a series of investments is envisaged.

3. OVERALL VIEW OF C-BAT

3.1 The C-BAT - as a product

The need for the SMEs to develop a CIM-based strategy based upon Business Goals, Marketing Goals, Manufacturing Methods etc. was quickly identified as being the most useful way to create a background for a small/medium-sized company evaluating CIM.



Indeed, for many SMEs the disciplines incorporated in the C-BAT methodology may provide a most valuable insight into their business.

Because C-BAT is a methodology supported by a software toolkit and not just a set of computer programs, users will benefit most out of the whole C-BAT concept through an education and training programme. Most appropriate would be seminars and tutorials at which the complete concept is introduced and guidebooks are given to describe how to use the methodology and 'tools'. In addition, exercises are performed by the attendees, and applied, where practicable, to their own operations, using the C-BAT methodology, with results presented on paper and as data on a PC.

The results of the exercises can be used as:

- the basis for more help by the tutors as consultants to the potential users

- as a starting point for those who wish to acquire the C-BAT software toolkit and use it on their own

- a feedback to improve the software toolkit

In this way it can be seen that C-BAT will be used effectively; firstly by the project partners and then by those who have been trained.

Extending the potential market in this manner is much more powerful than the sale of a set of programs and is more appropriate to achieve the strategic goal of assessment of CIM by SMEs throughout the EC.

3.2 C-BAT - How to use it

The underlying characteristic of the proposed methodology is the provision of structured decision-support in which the assessment of a CIM investment programme, as well as the monitoring of its eventual implementation, can be carried out.

It was decided that, provisionally, the C-BAT methodology would take an SME through a series of structured decision steps during the appraisal of a CIM investment project. The methodology is supported by a software toolkit, allowing the SME to generate and manipulate their information to support their decision process.

4. C-BAT - SOME DETAILS

4.1 C-BAT - The methodology

The methodology itself is initially envisaged as having **four phases**, each phase comprising a number of **modules** and each **module** being a **specific decision step** within the whole cost-benefit analysis.

Each module could be implemented by one or more techniques or methods, so specific choices will be made for those methods that can be implemented as the preferred tools and models.

Appendix 1 includes a set of system diagrams expressed in IDEF-0 methodology. The diagrams show, in a 'top-down' manner, the ways in which the **modules** are used in the Phases, the **types of information** that pass between modules and which **tools are appropriate to each module.** Thus, this system view forms the overall design of C-BAT.

The four Phases of the C-BAT methodology are:

- 1 Define Current Position (and make some predictions)
- 2 Do Conceptual Design and Evaluation
- 3 Make CIM Strategy and Investment Plan
- 4 Control and Adapt the Implementation Schedule

It is important to note that the methodology allows for multiple entries e.g. organisations who have completed an entire phase, or part of a phase, do not have to start from the beginning, they simply enter the methodology at the next logical step from where they are. In addition, the methodology is not rigid, in that it allows for feedback and a considerable amount of iteration, as can be seen from the 'looping' arrows in the diagrams.

Phase 1: Define Current Position

The purpose of this strategic phase is to encourage the SME to define its long-term objectives and its current achievements, and to make predictions for its future operating circumstances, taking into account the evolution of the market. Possible business problems are derived from these inputs.

The two possible outcomes of this phase are:

- the SMEs present structure and resources are performing adequately and will respond satisfactorily, in line with the predictions. At this stage, it is unnecessary to proceed further with the methodology
- the SMEs present structure and resources may or may not be performing adequately, but, is unlikely to cope and respond in line with the predictions

Sheet A.1.1 gives an overview of the activities involved in this phase and the information generated.

Phase 2: Do Conceptual Design and Evaluation

Having identified possible business problems, the main objective of this iterative phase is to propose a strategy to solve them. The strategy is then evaluated technically and, subsequently, it passes to an initial commercial feasibility study, because it may involve a substantial investment programme. Non-CIM strategies are not excluded, but the tools and models provided may be less appropriate because the main emphasis of Project 909 is to evaluate CIM.

Sheet A.1.2 gives an overview of the activities involved in this phase and the information generated.

Phase 3: Make CIM Strategy and Investment

In this phase, the emphasis is placed on investments in CIM, by definition of the objectives of Project 909. The CIM Strategy is analysed in detail, from business, as well as technical viewpoints. An implementation plan is produced and this is then analysed for its costs and benefits. The scheduling of this plan must be carried out taking into account factors such as:

- cash flow
- other company resources: personnel etc.
- integration of CIM elements or the architecture of CIM to be achieved

Sheet A.1.3 gives an overview of the activities involved in this phase and the information generated.

Phase 4: Control and Adapt the Implementation schedule

The accepted implementation plan from Phase 3 is put into action and is continually subject to monitoring and reviews. Actual costs and benefits are monitored against the plan by performing a series of audit exercises. A major company activity is to proceed with an education, training and awareness programme about implementing CIM for its staff.

Sheet A.1.4 gives an overall view of the activities involved in this phase and the information generated.

4.2 C-BAT - The Toolkit

Currently, a toolkit of ten tools has been identified (see 'Legend - Key' in IDEF-0 Diagrams). Those tools and models which are appropriate and feasible within the Project's resources will comprise the software toolkit. Some guidance may be given about criteria to apply in the selection of other techniques or methods.

The exact functionality of the tools is not available at the time of writing, but the list below gives an idea of the proposed functions for the ten tools that have been identified so far:

Objectives Database

- to hold information about Objectives, 'AS IS' and 'SHOULD BE' values etc
- to structure objectives
- to select objectives

Rating System

- to rank/order lists of objectives, targets, achievements, etc
- to assist in allocating a numerical figure to a qualitative parameter

CIM Element Database

- to hold basic information such as initial costs and benefits about CIM technologies in functional areas.

Gap Analysis

- to assess the differences between 'AS IS' and 'SHOULD BE' values

Financial Model

- to generate Balance Sheet and Profit-and-Loss accounts types of information

Investment Analysis

- to perform investment analyses using discounted cash flows etc.

Market Interface (Not a model of the company's market(s))

- to provide an interface to market analysis and to include market information in the subsequent analysis of CIM strategies.

Manufacturing Model Development Tool

- to structure and analyse the development and technical evaluation of the Manufacturing Model
- to use IDEF-0 methodology and include measures of technical performance
- to assess impacts of local change on global company performance
- to facilitate the integration of CIM elements

Cost/Benefit Database

- to hold data on the costs and benefits associated with proposals (to be used as input to the Financial Model and Investment Analysis)

Connectance Model

- to provide a broad-based qualitative link between technical performance and monetary/business aspects.

5. CONCLUSIONS

This paper has described a methodology for assessing CIM and the underlying need for a CIM strategy which must be derived from the business and market strategy. For an SME, the requirements to provide **guidance about potential 'early failures'**, adopt a **Market, Product, Technology approach** and make the **methodology and techniques easy to comprehend** are almost self-evident. However, the need to assess CIM within a 'company-wide' business strategy is less so and indeed may be novel to an SME.

CIM is not introduced into a company at a single instant in time - certainly not into an SME. A well thought-out CIM strategy introduces CIM over a considerable period of time which may be years. As such, **CIM is envisaged as a gradual, stepwise implementation of appropriate computer-based techniques**, subject to monitoring and reviews at either regular intervals of time or according to previously-defined technical or financial 'milestones'. Thus, any risks are greatly reduced as investment is spread over the period of time, and costs/benefits are monitored regularly.

This methodology is supported by a set of tools which are currently being developed in the context of the project. The next phase of the project sees the software implementation of these tools in the context of the methodology.

In order to increase the Project's industrial impact, it is intended to set up a demonstrator during the first quarter of 1988. The aim of this demonstrator is to implement and test the prototype C-BAT. In light of its expected relevance to industry, the partners in the Project have expressed their intention to continue the industrial application of the C-BAT beyond the end of the project.

6. ACKNOWLEDGEMENTS

We should like to thank the members of the project consortium for their work in the project upon which this paper is based.

Partners :

AMTRI	TELEX :	668020 AMTeC G	Fax (UK):	0625 34964
BIBA		174212196 BIBA+D		
CIMAF		63423 ISTUTL P		
DTH		37529 DTHDIA DK		
T.I.		33416 TIDK DK		
Mentec		627613 MENT G		
		93309 MENT EI		
WTCM-CRIF		25393 CRIF B		

Type of manufacturing	Percentage
Line	12.9%
Batch	60.7%
Jobbing	26.4%

Table 1: Type of manufacturing

Work	Percentage
To order	71%
To stock	29%

Table 2: Work : order or stock

Percentage of number of companies	IN USE			PLANNED	
	Manual methods	Computer based methods	Linked by computer database to other activities	Computer based methods	Linked by computer database to other activities
Administration	13.6%	45.8%	27.1%	6.8%	10.2%
CAE	35.6%	28.8%	15.3%	6.8%	10.2%
Manufacturing management systems	20.3%	33.9%	18.6%	13.6%	13.6%
Shopfloor systems	61.0%	13.6%	3.4%	8.5%	8.5%

Table 3: Computers in business

Configurations	Percentage of number of companies
CNC/DNC	72.3%
Manufacturing cell	40.0%
Robotic cell	32.2%
FMS	13.8%
CAD-elements	58.5%
CAM-elements	50.8%
CAD/CAM configurations	48.4%
Advanced material handling	15.6%
Production monitoring	49.9%

Table 4: 'Technical' computing in use

Range of costs (1,000 ECU)	Percentage of the total number of companies
0 – 500	60.7%
501 – 1,000	21.4%
Over 1,000	17.9%

Table 5: Cost of introducing CIM elements

Factor	Percentage of the number of companies
Corporate strategic plan	62.3%
Financial constraints	31.1%
External request	20.0%

Table 6: Influencing factors on CIM implementation sequence

BENEFITS		PERCENTAGE IMPROVEMENT			
		0% – 30%	31% – 60%	61% – 100%	> 100%
Reduction in material costs	P	87.5%	12.5%	0.0%	_____
	A	87.5%	12.5%	0.0%	_____
Reduction in labour costs	P	77.8%	22.2%	0.0%	_____
	A	74.0%	25.9%	0.0%	_____
Reduction in stock	P	75.0%	16.7%	8.3%	_____
	A	83.3%	16.7%	0.0%	_____
Reduction in work in progress	P	80.0%	20.0%	0.0%	_____
	A	100.0%	0.0%	0.0%	_____
Reduction in tendering time	P	57.1%	14.3%	28.6%	_____
	A	42.9%	28.5%	28.6%	_____
Reduction in delivery time	P	66.7%	26.7%	6.6%	_____
	A	73.3%	20.0%	6.7%	_____
Reduction in product lead time	P	66.7%	25.0%	8.3%	_____
	A	75.0%	16.7%	8.3%	_____
Increase in product mix/variant	P	100.0%	0.0%	0.0%	0.0%
	A	100.0%	0.0%	0.0%	0.0%
Increase in machine utilisation	P	85.0%	15.0%	0.0%	0.0%
	A	90.0%	10.0%	0.0%	0.0%
Total reduction in product costs	P	89.5%	10.5%	0.0%	0.0%
	A	84.2%	15.8%	0.0%	0.0%
Increase in sales	P	77.8%	11.1%	0.0%	1.1%
	A	77.8%	11.1%	0.0%	1.1%
Increase in operation profit	P	66.7%	0.0%	33.3%	0.0%
	A	66.7%	0.0%	33.3%	0.0%
Increase in quality	P	100.0%	0.0%	0.0%	0.0%
	A	100.0%	0.0%	0.0%	0.0%

Table 7: Benefits – Predicted and Attained

MAJOR PROBLEMS OF CIM INVESTMENT	
ECONOMIC	Difficult to evaluate Payback period Limited funds
MANAGEMENT	Lack of available tools Difficult to manage Lack of commitment in management
TECHNOLOGY	Lack of system understanding Lack of software Lack of skilled people High customization effort

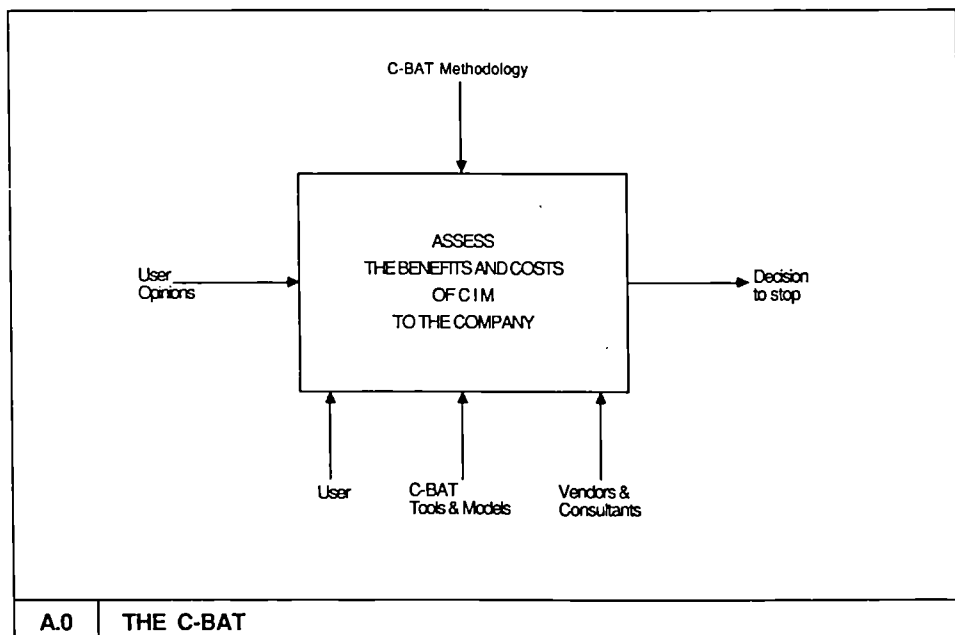
Table 8: Major problems of CIM investment

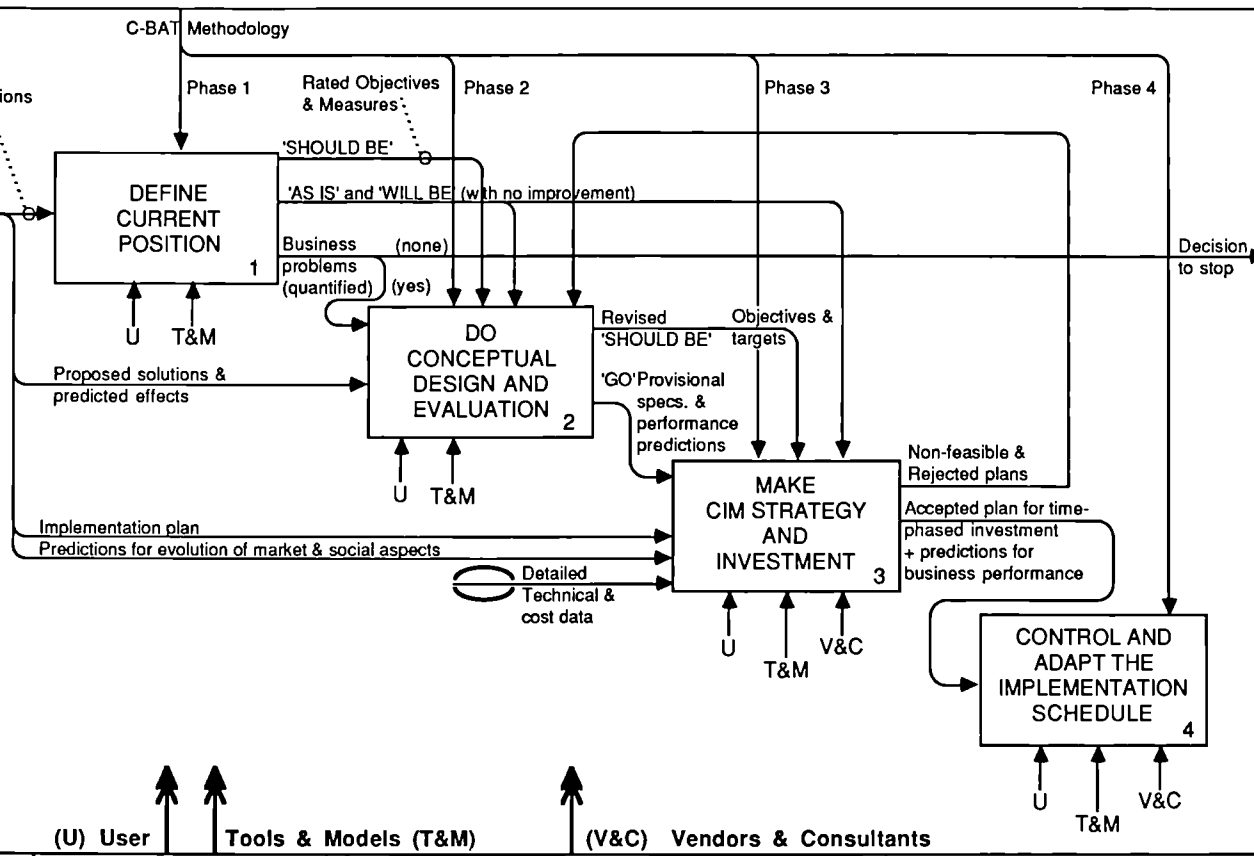
Economic evaluation method	Percentage of companies that:	
	Had used each method	Would use each method
– Cash flow forecast	18.2%	25.0%
– Accounting rate of return	10.6%	13.6%
– Payback period (non-DCF)	21.2%	28.8%
– Internal rate of return (DCF)	15.2%	24.2%
– Net present value	22.7%	21.2%

Table 9: Financial criteria applied and 'to be' applied (some use more than one criterion)

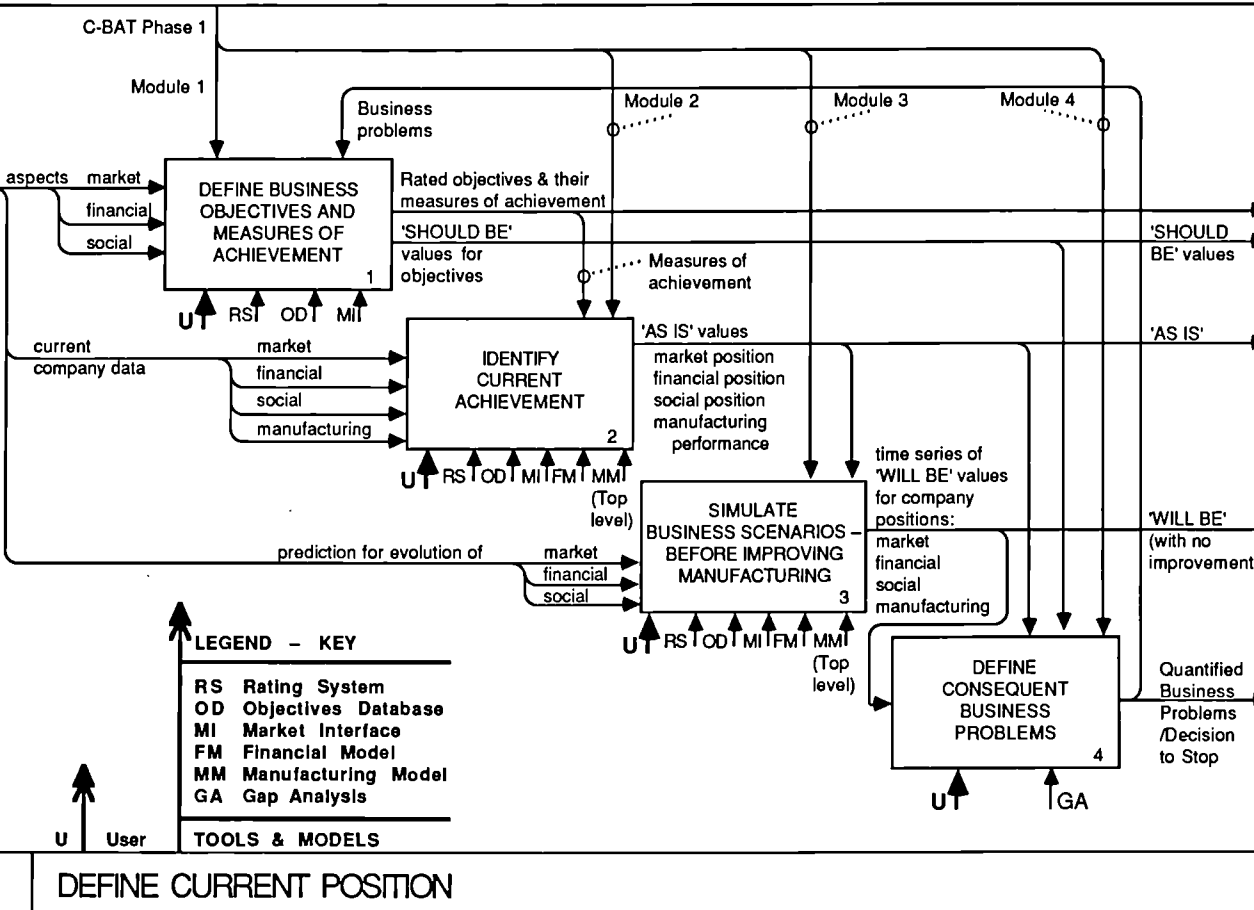
APPENDIX

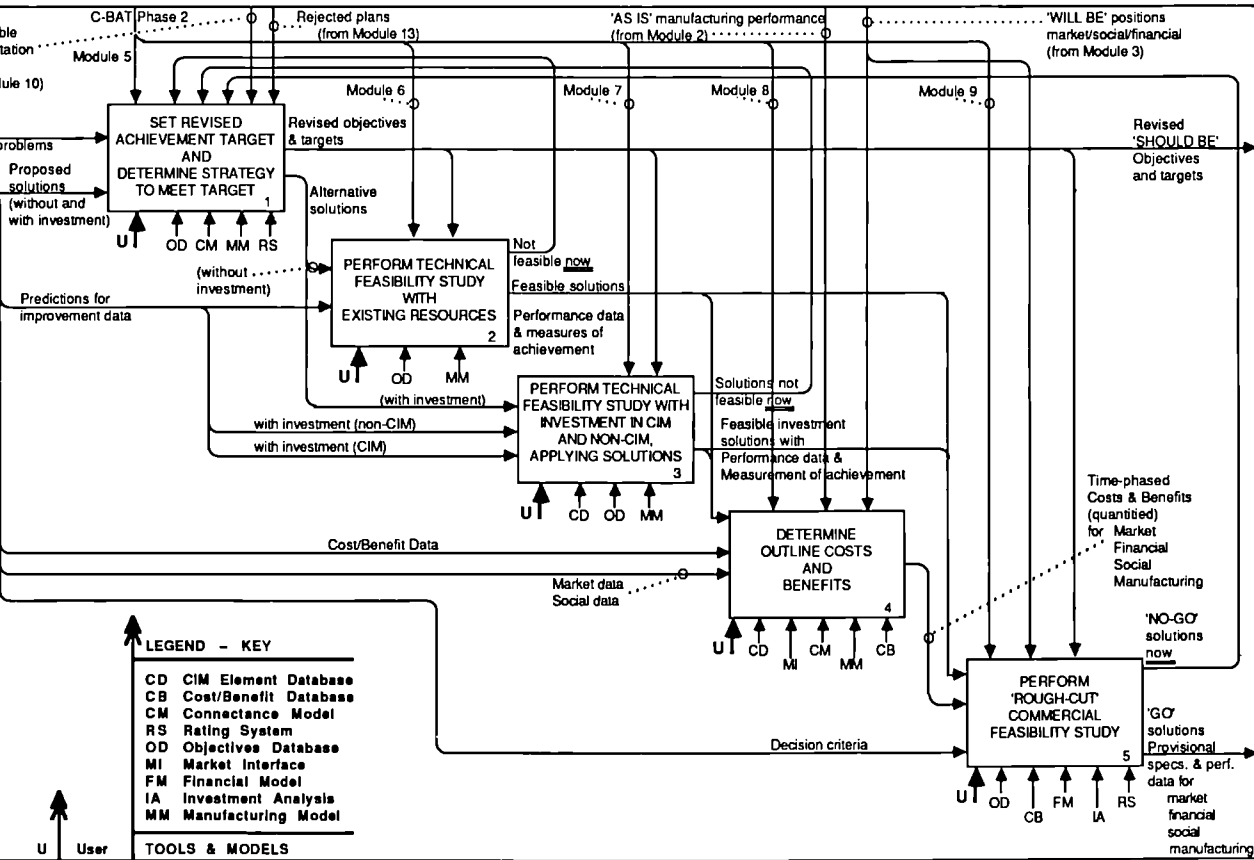
IDEF-0 diagrams of C-BAT



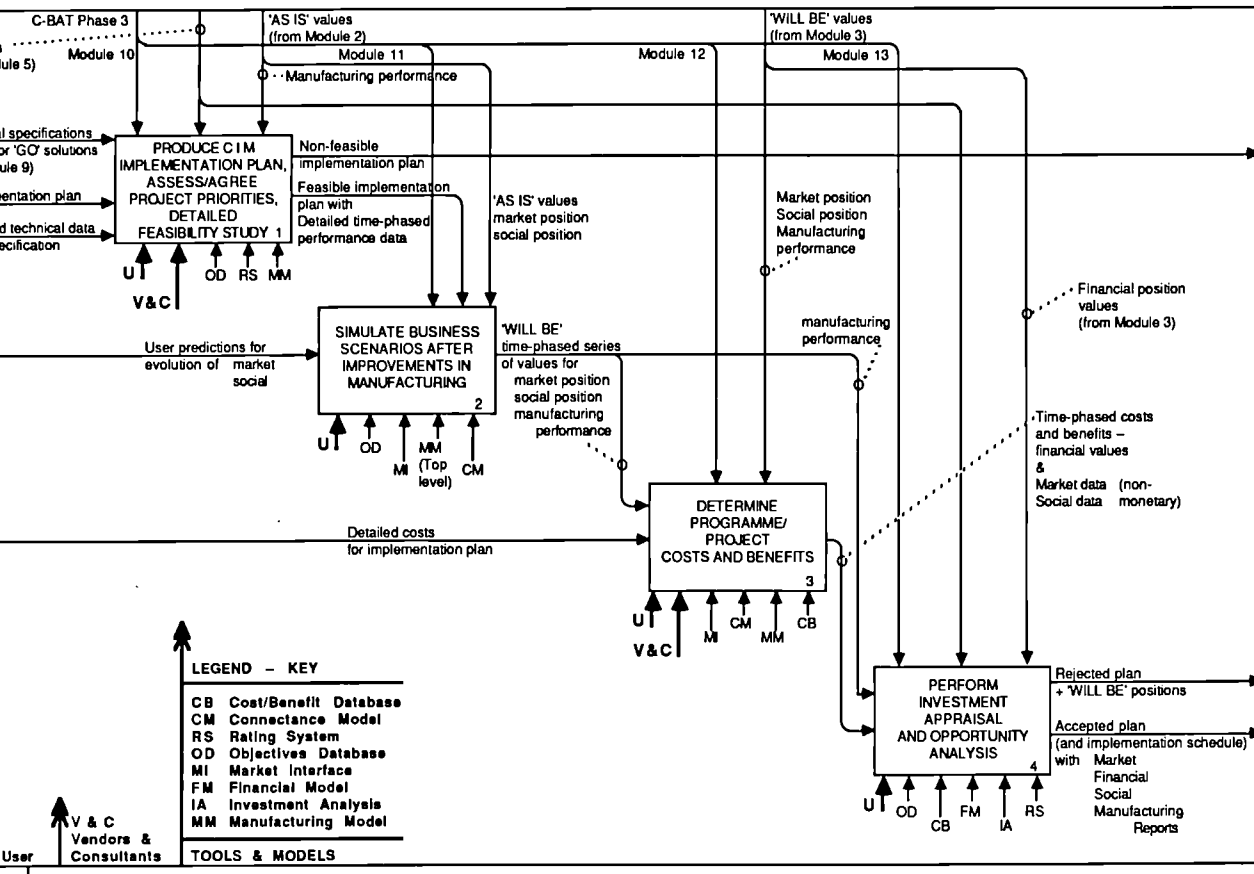
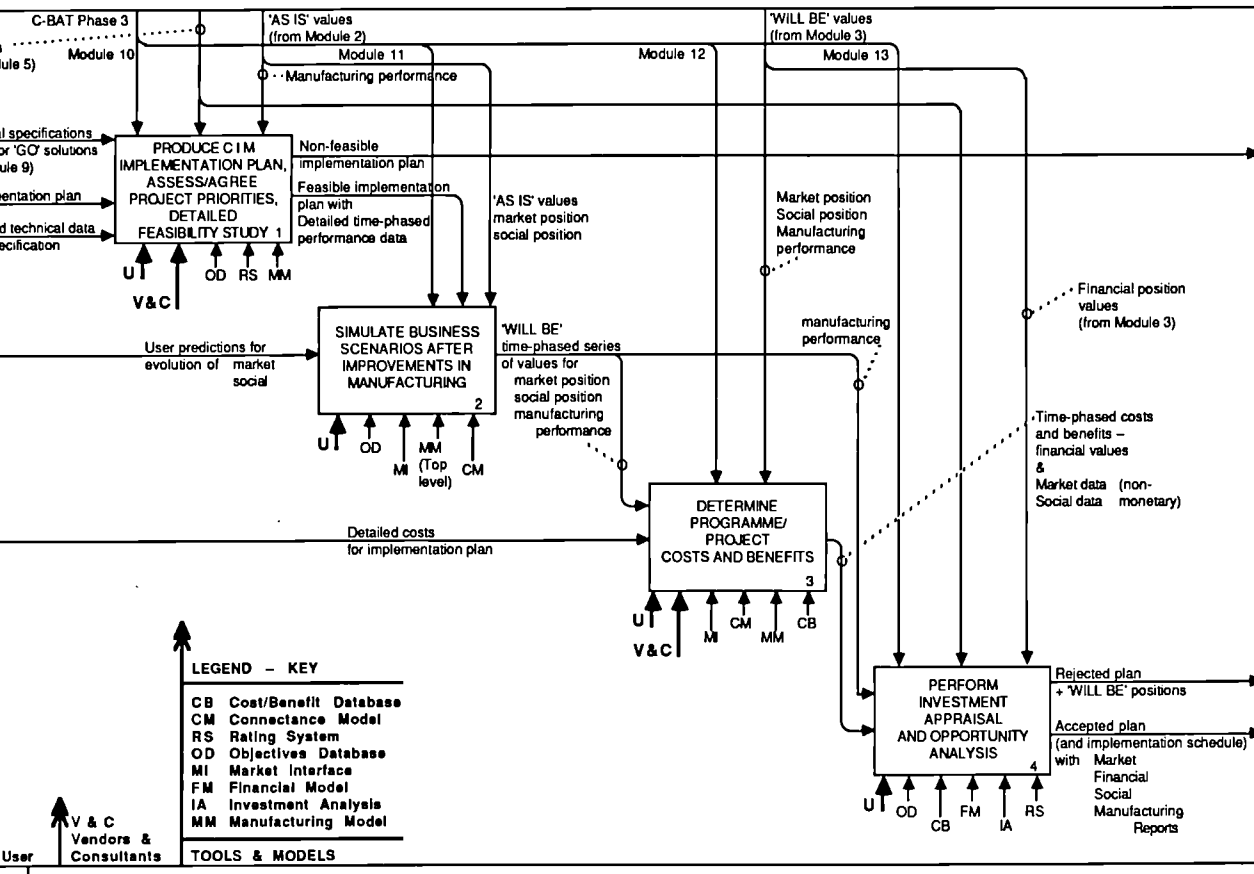


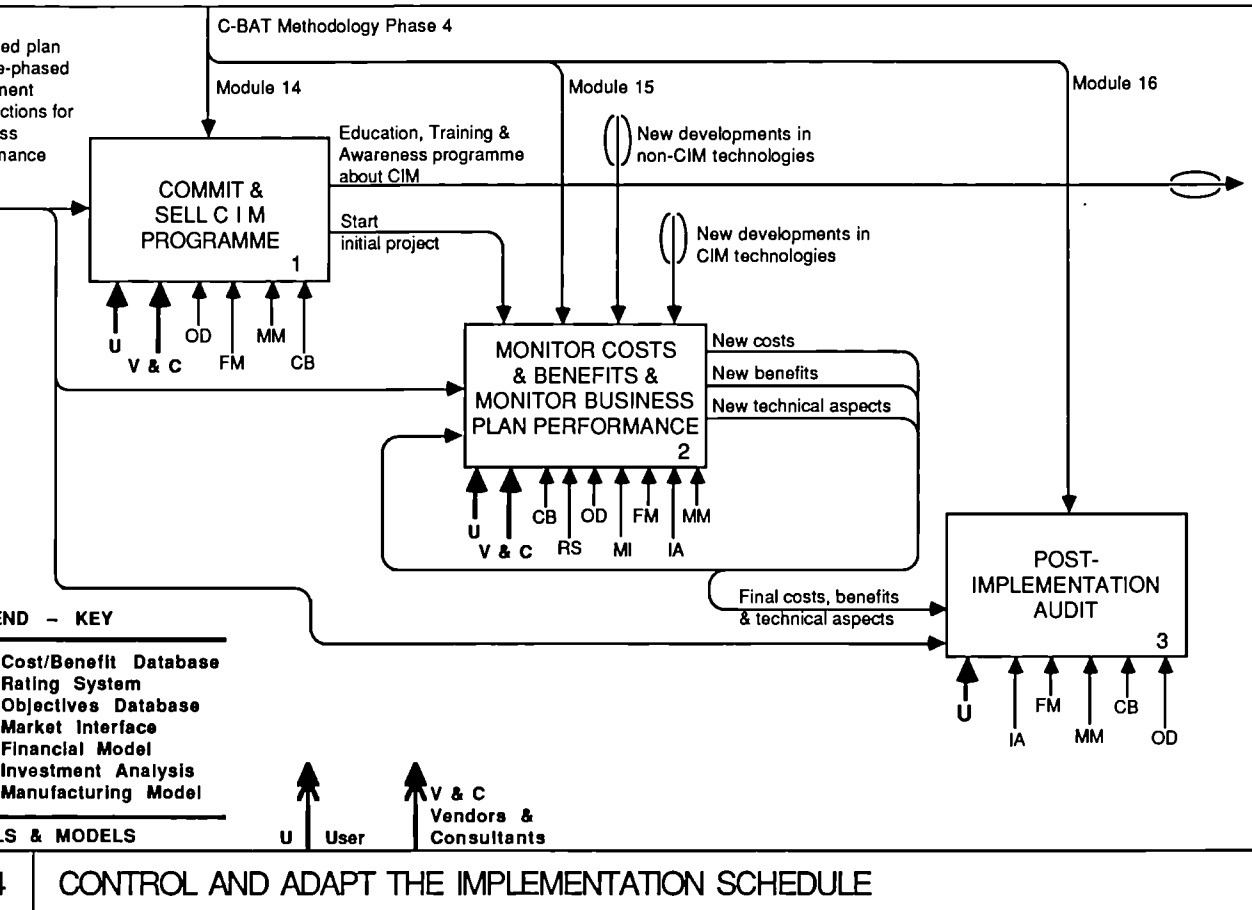
ASSESS THE BENEFITS AND COSTS OF CIM TO THE COMPANY





2. CONCEPTUAL DESIGN AND EVALUATION





Project No. 812

ESPRIT PROJECT N. 1217 (812)

EXPERIMENTAL CENTER FOR SYSTEM INTEGRATION IN CIM

C. BORASIO, P. MURCHIO (ELSAG - GENOVA - ITALY)

R. GROPPETTI (POLITECNICO DI MILANO - ITALY-)

D. SOLA (AERITALIA G.V.C. - TORINO - ITALY)

ABSTRACT

The paper present the main aims and in-progress results of the research project "Experimental Center for System Integration in CIM" N. 1217 (812) as part of the ESPRIT program.

The objectives of this project are the study and the implementation of an experimental CIM (Computer Integrated Manufacturing) center for the development of tools, evaluation of system integration concepts and test of prototypes.

Starting from an open kernel configuration and in connection with the other ESPRIT CIM architectural projects, this system will be implemented and expanded to meet the needs potential users. The system will provide a CIM environment, in the area of flexible mechanical manufacturing, suitably open to integration, for the development and evaluation of different subsystems, components, architectures, communications and system approaches to CIM technology.

1. INTRODUCTION

An analysis of the international research, development and implementation in the area of CIM (Computer Integrated Manufacturing) both in industry and in research institutions, shows a large variety of efforts and initiatives, and a general state of confusion among suppliers and user.

This situation is due to the lack of common reference models, data processing and communications strategies, standards for subsystems and components integration in a heterogeneous and distributed control manufacturing environment.

This state could be brought to an end by international and inter-companies initiatives, which lead to a coherent, well evaluated and demonstrated, and internationally approved set of CIM architectural and interfacing standard, allowing easy and reliable CIM integration of subsystems and components in a multi-vendor environment.

However in the perspective of the development of a set of CIM standards it would be greatly opportune to have a common reference point, where European efforts and development for integration could interact, and possibly coalesce. An important reference point could be an experimental integration center, open to heterogeneous and progressive integration.

For the development of such a center the Commission of the European Economic Community placed, as a part of ESPRIT research program, the project N. 1217 (812) on the study and development of an "Experimental Center for System Integration in CIM" to a Consortium of European partners, which includes qualified industrial companies, universities and research institutions as follows:

ELSAG S.P.A.	(I)	- Prime Contractor
PHILIPS & MBLE Ass.	(B and NL)	- Contractor
SESA S.A.	(F)	- Contractor
AERITALIA G.V.C.	(I)	- Contractor
POLITECNICO DI MILANO	(I)	- Contractor
WZL TH AACHEN	(D)	- Contractor
DEA S.P.A.	(I)	- Subcontractor
ITALCAD S.P.A.	(I)	- Subcontractor
IMU - CNR	(I)	- Subcontractor

Purpose of the Consortium is to implement, in co-operation with the ESPRIT architectural projects, an experimental center within an existing mechanical engineering laboratory.

The basic module of the center will be arranged on the basis of the existing facility which is being made available by ELSAG to the project.

2. MAIN OBJECTIVES OF THE PROJECT

The research project N. 1217 (812) has the following main objectives:

- To build up an experimental center for the study, development and testing of system integration concepts and tools in a Cim environment; the center will demonstrate both the integration of fundamental manufacturing functions from product design stage to production planning and control, and the physical integration of manufacturing process, inside of a typical industrial mechanical manufacturing environment.
- To provide a system, with the necessary basic hardware and software facilities, where tools, subsystems and prototypes, developed in the ESPRIT Project in the CIM area and/or provided by the partners, can be integrated, tested and refined in an environment near to the real production one.
- To improve the capability of European industry in implementing complex and extended Flexible Manufacturing System, using heterogeneous equipment and modules from different vendors integrated in an open architecture.
- To support European standardization efforts in validating the emerging communication standards and CIM reference models.
- To allow, on a practical basis, through the implementation of the center, the testing of prototypes developed within ESPRIT.
- To demonstrate the feasibility of a center open to the progressive integration of different factory function and manufacturing processes.
- To demonstrate that CIM system suitably designed and developed in its software and hardware aspects, allows the portability of key common elements to other facilities, and the migration path toward emerging standards.

3. PROJECT METHODOLOGY AND WORKPLAN

In order to achieve the project objectives, a methodology was adopted starting from context analysis and requirements identification.

This methodology is based on three fundamental activities, that have been planned from the beginning:

- Analysis and selection of a set of industrial processes and manufacturing typologies which present interest or general strategic value and market impact in the European industry for CIM application and identification of the constraints, which limit

CIM implementation in the selected areas.

- Specification, development and implementation of hardware and software components for the center, including a range of manufacturing equipments allowing the implementation of a predefined set of manufacturing processes.
- Close interaction with other ESPRIT projects so that system architecture and communications could consider the results of the relevant ESPRIT project like N.688 (CIM-OSA "Open System Approach") and N. 955 (CNMA "Communication Network for Manufacturing Applications") [6, 7, 8].

For the development of the project the activities are grouped in eight main tasks identified as follows:

- Definition of the functional context of the center and external relationships.
- Identification of manufacturing typologies.
- Study of technological processes.
- Study of the material handling/transport subsystem.
- Study of data structure and DBMS.
- Study of communication subsystem.
- System development and set-up (installation, integration, testing and validation).

In addition to these technical tasks: other activities are related to external connections, and to project management and control.

The four years project is carried out initially over a three years contract, starting from 1986. The workplan presents different phases as shown in the diagram in Fig.1.(see next page)

Project Plan

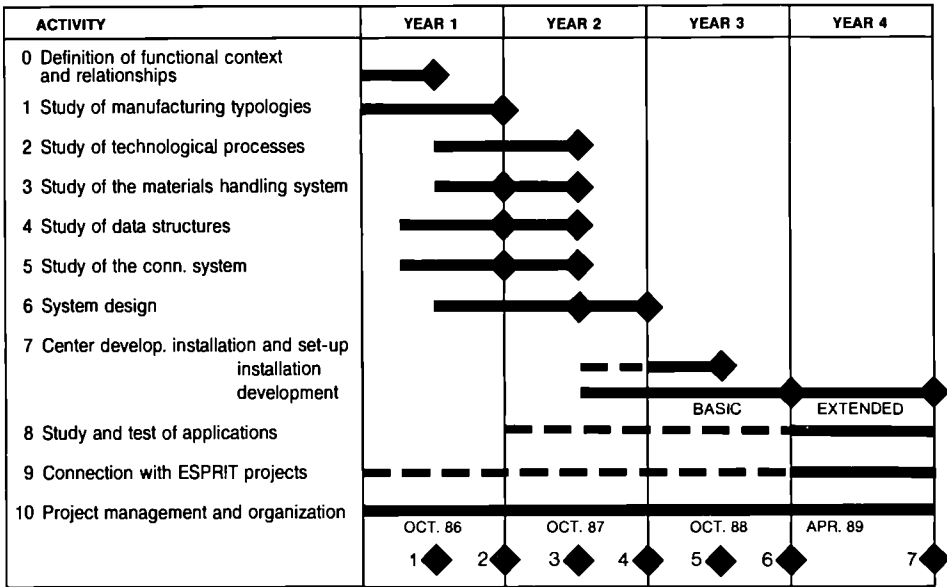


FIG. 1

4. PROJECT ADVANCEMENT AND RESULTS

At the end of preparatory phase the objectives concerning the definition of the features and the specifications of the center, including the architecture and applications scenario have been achieved.

The main results are described in the following paragraphs.

4.1. MANUFACTURING TYPOLOGIES AND CENTER LAYOUT

The study of manufacturing typologies has analysed products, related technological processes and manufacturing systems of specific interest for CIM applications and suitable to be implemented. Starting from the analysis of the state of the art a survey was done on existing CIM/FMS installations both in industry, for systems with production mission, and in research, for systems with experimental/demonstration mission. Then the study has analysed, mainly using market and public domain studies and specific experience of the industrial partners, the main manufacturing typologies in order to extract and compare information on products, processes, manufacturing systems, presenting strategic relevance, both from suppliers and users point of view.

Due to their relevance besides the industrial installations, the main experimental/demonstration systems, installed or in development, were analysed [1, 5].

Therefore some suggestions and criteria were established in order to reach the preliminary definition of a possible starting FMS configuration as an expandable integration and test bed facility, suitable to further expansion having a strategical impact among the potential users.

The result of the mentioned analysis are depicted in the following table, and has been used for defining the operational features of the Center:

- Manufacturing Sector:

80 + 90 % FMS installation are in mechanical/transport industries.

- Part Typology:

Prismatic (70 %) and rotational (20 %) parts.

- Material Typology:

70 + 80 % Cast iron, Aluminium Alloy, Steel.

- Machine Typology (in term of system application):

80 % Machining centers.

30 % Turning machines.

20 % Inspection machines.

15 % Washing machines.

20 % Milling machines.

20 % Special machines.

- Parts Dimension:

The maximum number of workpieces is inside a cube of 500 mm.

The FMS application future trend is for smaller and lighter workpieces.

According to the purpose of the center, some fundamental cells/work-stations were selected for the starting kernel:

- Machining cells for both prismatic and rotational parts.

- Assembly cell.

- Inspection station.

- Washing/workstation.

- Special/innovative process cell (suitable for expansion).

and the following subsystems:

- Warehouse (parts, tools, fixtures).

- Tool room and tool presetting.

- Material handling and transport subsystem.

- Load/unload stations.

All the different cells have to be linked physically by the transport subsystem and have to communicate and share data.

The morphological characteristic of the workpieces and the complexity of the products have been selected in order to cope a wide range of applications.

The mentioned study and the analysis of the operational needs have lead to a starting configuration of the center as shown in the general layout of Fig.2. The configuration takes also into account the budget constraints.

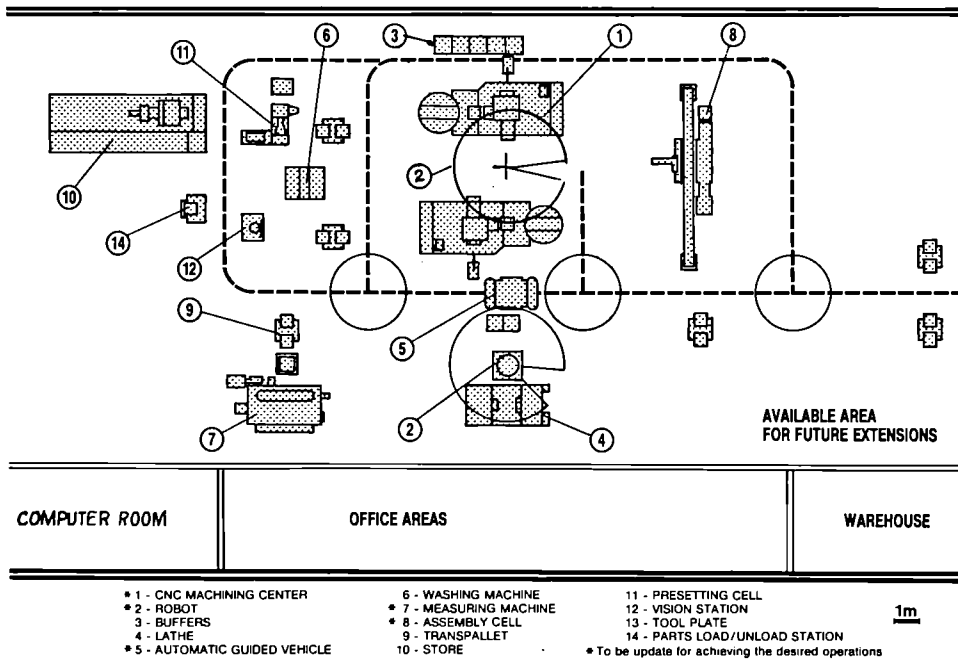


FIG. 2

4.2. DATA STRUCTURE AND COMMUNICATION SUBSYSTEM

Data processing and homogeneous, integrated industrial communication system, are widely recognized as fundamental issues of any industrial automated environment.

A single Data-Base Management System may support several distinct data-base, that can be entirely unrelated, or linked as is often the case between design data-base and production data-base. On the other hand, a single data-base, cannot be distributed on various, heterogeneous system without extensive software develop-

ment, therefore the use of a single data-base for the whole center is not realistic. As a consequence, the selected data organization of the system can be described as follows:

- Subsystems, homogeneous from the functional point of view are groups of functional entities clustered around a local data-base and tightly coupled by it.
- Different subsystems are loosely coupled by means of a data exchange through a data-base server where relevant data are stored in a vendor independent standard format. The exchanges among subsystems are made possible by software interface layers.

Data communication architecture can be seen as the fundamental requirement of the experimental Center. In fact purpose of the project is to implement an open communication subsystem, according to the communication standard emerging from MAP (Manufacturing Automation Protocol) and CNMA, and in particular to implement the RS-511 (MMS) protocol as a common language for communication among the different equipment [8].

The current strategy of implementation of the communications inside the system is to make a design in accordance with the emerging standards and to start from the implementation of a transparent structure at "application level" using interim solutions where standard facilities are not available (e.g. at cell level where private network will be used).

These interim solutions are conceived for an easy integration in multi-vendor environment.

The outline of communication networks is shown in Fig.3.

4.3. FUNCTIONAL/IMPLEMENTATION MODEL

A functional analysis of the whole system by means of the IDEF-0 (SADT like) methodology has been performed.

The final results of this analysis, together with the manufacturing typologies requirements study, has been the identification of the required functions and activities to be implemented on the Center:

- To design parts.
- To define process plans and part programs.
- To define and select tools and fixtures necessary for the manufacturing operations.
- To plan production.
- To give production orders.
- To machine parts, starting from raw material, following a suitable sequence on the workstations.
- To test and measure parts.
- To assemble parts.
- To store, retrieve, transport parts and tools.

Besides the typical features of a FMS in the field of small/medium

batch size discrete part manufacturing the Center functionality will include also these other features:

- Integrated tool management.
- On line quality control on the product with real time feed-back to the process.
- Assembly operations integrated in the process cycle.
- Use of a vision subsystem to support inspection at system level.
- Production scheduling by means of an expert system.

The results of previous work, together with the definition of the data that have to be exchanged, are the system specification describing the framework for the subsequent software development activity.

Starting from this work it has been possible to define an homogeneous and comprehensive hardware and software architecture (see Fig.3). In the "implementation model" for each function a corresponding package has been defined and the responsibility for its development has been assigned to the partners. Several modules of the software architecture will be made available to the project by the partners or will be obtained from the market.

(see next page).

4.4. CONNECTION WITH OTHER ESPRIT PROJECTS

According to the project plan, in order to reach the objective to establish permanent connection and co-operations with other ESPRIT research projects, the relevant ESPRIT projects were selected following the criteria of technical complementary and practical co-operation feasibility.

Special attention is devoted to the complementary project N.1199 "Human Center CIM System" for what concerns all the issues related to the human interface of CIM components and subsystems, and to the projects closely interconnected as the project N. 688 on the development of an open system architecture for CIM [6], and the project N. 955 concerning communication network inside a CIM environment [8], and the project N. 322 ("CAD Data Exchange").

4.5. APPLICATION SCENARII

The experimental center has the main aim to demonstrate system integration and evaluate and testing CIM subsystems and components in the field of discrete part small/medium batch manufacturing, oriented to metal cutting processes.

Therefore during the design and development of the system it is required to identify analyse the applications of potential users, to be interfaced and integrated.

For this purpose hardware, software and user interfaces will be designed in order to be standardized and open at the maximum extent.

Due to the project focus on CIM integration in a context of flexible manufacturing, the applications will be mainly in the areas

Experimental Center architecture (preliminary)

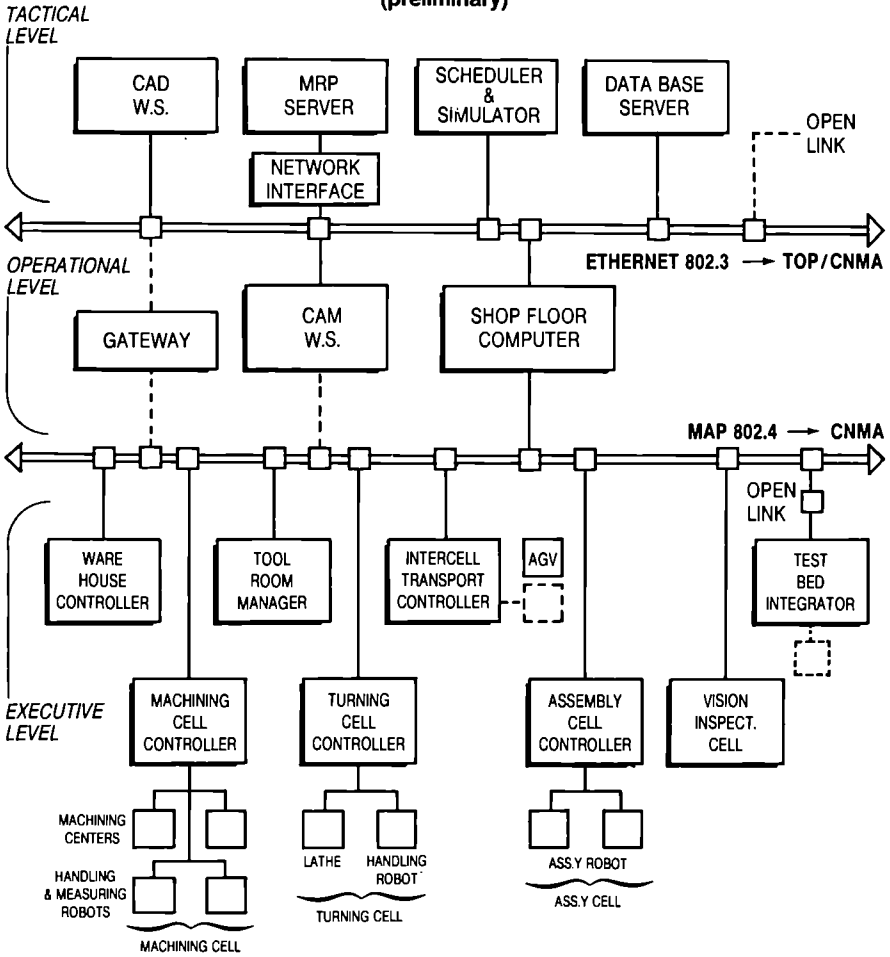


FIG. 3

of product and process design, production planning and control, operation control of cell/equipment.

The experimental Center will allow also the emulation of processes, components and subsystems at different levels (e.g. virtual cells), whose behaviour can be integrated into the system, without their physical implementation in the actual configuration. Due to the presence of different cells and to the possibility to integrate and link further cells and equipment, the Center allows flexibility in operation on different workpieces and with different technological processes, even if only basic processes have been selected and implemented into the starting configuration. Regarding to the network point of view, different kinds of station and controllers could be accepted and tested.

There is also the prevision to link a Test Bed Integrator to the network that is a user interface to test and validate different modules or subsystems without interferring with the operations of the system.

Other applications are related to the assessment of the center architecture in order to demonstrate its capability to represent different class of manufacturing entities and typologies, its capability to fit within generic and advanced architectures. Therefore for the communication requirement at the different architecture levels (factory, shop, cell, unit) and for the future requirements, the extended use of CNMA standards and networks is foreseen.

The control architecture and the physical layout of the center could demonstrate the capability to integrate new cells and new work-units, and to assess alternative cell and work-unit models. The Center, could support several experiments regarding the link between design and production. In the area of product design and process planning the use of standard neutral format for product data definition, for both machined parts and assemblies, could be demonstrated. In the area of production planning, and scheduling, modules, able to transform MRP orders into executable cell orders, could be validated.

Finally the experimental Center can assess the application of monitoring subsystems for supervising the manufacturing process. The general modalities to access the demonstrator Center by different users (e.g. ESPRIT project developers), are:

- To achieve a technical and financing agreement among ITTF, application proposer and Center responsible.
- To establish a detailed program for experiment.
- To define interface requirements and modalities.
- To negotiate the availability of the Center facilities, the relevant costs and financing aspects connected to the technical services required by user, in order to carry the application study.

BIBLIOGRAPHY

[1] WECK, M., Dern, U., Integrated Manufacturing and Assembly, Annals of the CIRP, Vol. 34/1/1985.

[2] KIMURA, K. Tsukuba FMSC Test Plant, Proc. of the 5th Conf. on Prod. Engin., Tokyo, 1984.

[3] KIMURA, M., OZAKI, S., SODA, C., YOSHIDA, Y., TATSUE, Y., KANAÏ, M., Flexible Manufacturing System Complex Provided with Laser, Proc. of the 5th Conf. on Prod. Engin., Tokyo, 1984.

[4] SIMPSON, J.A., HOCKEN, R.J., ALBUS, J.S., The Automated Manufacturing Research Facility of the National Bureau of Standards, Journal of Manufacturing Systems, Vol.1, N.1, 1982.

[5] JONES, A.T., MCLEAN, C.R., A. Proposed Hierarchical Control Model for Automated Manufacturing Systems, Journal of Manufacturing Systems, Vol.5, N.1, 1986.

[6] HUYSENTRUYT, J., CIM-OSA a Computer Integrated Manufacturing Architecture Based on the open System Approach, Proc. of the second Int. Conf. on Computer Applications in Production Engineering Copenhagen, 1986.

[7] YEOMANS, R.W., CHOUDRY, D., DEN HAGEN, P.J.W., Design Rules for a CIM System, North-Holland, Amsterdam, 1985.

[8] CNMA, Implementation Guide, Revision 1.0, ESPRIT Project N. 955, 1986.

Project No. 688

CIM-OSA STRATEGIC AND MANAGEMENT ISSUES

by

B. Lorimy
CAP GEMINI SOGETI, Paris (FR)

P. Russell
ICL/STC, Stevenage (UK)

1. INTRODUCTION

The AMICE Consortium - ESPRIT project 688 - has been working on the development of a Computer Integrated Manufacture Open Systems Architecture for about two years.

At this point of the project it appears useful for the 19 partners as well as for the community gathered under the ESPRIT banner to assess the current achievements and the main expectations concerning the major milestone to come.

As major manufacturers, users, software houses and research institutions throughout Europe the members of the Consortium are quite aware of the difficulties of the task they have been tackling, as well as the strategic advantages they feel they will derive from the project. It is the right time to evaluate and explain both.

On such issues managerial and technical views are deeply interleaved with each other. Implementing the technology of to-day might be a purely technical question, although even a programme like MAP is now constrained by more general problems than just the choices of technical standards by which it got started.

When, in the attempt of catching up with the international competition, especially from Japan industry in the manufacturing area major conceptual steps are looked for, as in the AMICE project, business modelling and technical questions are one and the same set of concerns for management.

Although ESPRIT is a precompetitive research programme, in any project the investments made by partners and the financial support from the Commission can only be justified ultimately by actual tangible results in terms of business.

In other project where technical steps are looked for, implementation is easy, benefits are more predictable at shorter term;

The purpose of this paper is to show - in line with the CIM evolution - that

- AMICE has been concentrating on medium term results in long term perspective very remote from any dream but also from the assembling of trivial technologies of today.

- It is not an act of faith but a technical and strategical realization based on the experience of the participants as well as the current trends of the IT technology, especially in the software engineering domain.

It is also by reviewing some of the major concepts of CIM-OSA to explain the use of such an architecture not only in new CIM systems to be developed but as importantly in the evolution of current manufacturing systems.

Definitely at this level it is legitimate during the Technical Week to explain how management considerations orient the technical work of such a project of crucial importance for the industry.

2. MAJOR ACHIEVEMENT: A COMMON VIEW OF CIM

AMICE project has issued on 31st of January 1987 as an official milestone its "CIM-OSA Specifications Phase Deliverable" document called in short CIM-OSA S1.

This document has been backed by an impressing set of internal documents which are illustrating the draft architecture that has been produced. In addition to general contributions such as formal Representation of CIM user Industry, General Properties of Target CIM System, these internal documents include a part devoted to formal references of the concepts manipulated during the design. Beyond documents, the partners realize that their main achievement has been sofar a common view of CIM, formalized whenever it was possible, that they all share and will propose as a Reference to the community in due time i.e. when their common conceptual construction has been sufficiently "checked out" by real cases. Only at their level - 19 companies - it was not so easy. Computer Integrated Manufacturing is still many things for many people. Starting from the core of Manufacturing - Product Development, Production Planning and Production - CIM is evolving to become a synonym for total integration of the manufacturing enterprise. However the word "integration" is not understood consistently. Computer Integrated Manufacture even less as many people in the manufacturing enterprises have only a limited view of what computers can do for them beyond isolated applications. Points of view also differ according to the business objectives that are pursued, and levels of management responsibilities within an organization. CIM users and CIM suppliers, manufacturers and service provider, researchers and developers are not sharing exactly the same understanding, although there is an intuitive consensus in broad and vague terms on the necessity of better - computer based - communication within a manufacturing company and also

beyond, between such a company and its partners. Some of the various ways of understanding CIM are reflected in figure 1 which shows the most frequently used associations with the term CIM.

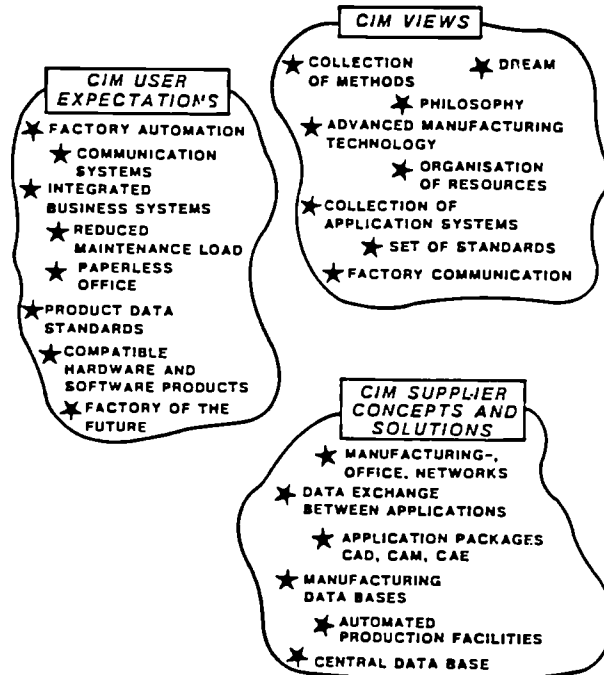


Figure 1: The CIM Environment

More importantly the concerns about CIM is not exactly the same at various levels of management.

Corporate management of user company expects far more than just an increase of productivity or quality. The corporate optimum is no longer the sum of local optima. Other intangible objectives such as independence with regard to vendors, minimum risk strategical paths etc. are more and more included in top management expectations. Middle management are also expecting benefits from CIM at their level while often resisting to the most extreme forms of CIM which would challenge their jobs.

The achievement of the AMICE has been to overcome the main difficulties, eliminate the main discrepancies along the route towards a universally accepted integrating mechanism respecting the infinite variety of the specific situation.

The most difficult problem for AMICE to overcome was the cultural differences between the manufacturing specialists and the computer software specialists. The Consortium wanted to avoid two extreme cases:

- the manufacturing experts blocking solutions that they would not understand taking advantage of their "concrete" experience
- the computer specialists designing architectural solutions without any grasp on the real world of the shop floor.

Such a problem does not exist in software engineering area where can be found most of the concepts contemplated and used by AMICE, but with the advantage of an homogeneous culture of all specialists.

The complexity of the subject - the manufacturing enterprise itself - cannot easily be tackled in scientific terms. The science of manufacturing organizations is a social service. People are of the essence no easily reducible to common engineering practices on the shop floor, as well as in the business in general.

This explains the problem of consensus about the meaning of CIM. Product, manufacturing and information technologies, production and market logistics, investment policies are only some of the many dimensions of the enterprise complexity. Many interrelations exist between these aspects and all must be considered simultaneously. Almost all are also time dependant.

To reduce this complexity a structuring scheme has to be provided which allows better visibility of the main characteristics of the manufacturing enterprise and provides the means to integrate lower complexity parts into the system.

Encapsulation of "details" in higher level constructs, hiding the real system complexity from the user is the driving idea of CIM-OSA, as it was for a much more limited problem - computer communications - in the OSI model, as it is to be in the most advanced forms of software engineering.

3. CIM-OSA IN PERSPECTIVE

The basic concepts developed by the AMICE project will guide further developments of the Open Systems Architecture which will provide guidelines for CIM users and CIM suppliers. CIM-OSA will help the users in designing, selecting, implementing and operating CIM systems, CIM subsystems and CIM system components. It will guide the suppliers in the design and development of such CIM systems, subsystems and components that will have a ready market. In order to provide a local point for the convergence of ideas, CIM-OSA describe not only the

strategic aim for CIM, but also how the industry can progress incrementally towards that long term aim.

As a matter of fact, CIM has to be understood as an integration process leading over time to an integration of the manufacturing enterprise.

Along this process, different levels of integration can be seen, where the emphasis is progressively systems (Physical Integration) to the Applications Integration i.e. the supply and removal of information, communication between applications and users (human being as well as machines) and with the system itself and ultimately to the Business Integration. Only at this level will it be possible to define adequate requirements for Applications and Physical Integration and only the understanding of the business needs from an overall system point of view will allow the definition and development of the right application for the individual enterprise, thus bringing the real benefits that company management is commonly associating with CIM.

CIM-OSA work is a step forward in that ultimate direction:

- it was not the intent for CIM-OSA to reinvent schemes, such as OSI or MAP which are dealing with Physical Integration in a bottom-up approach, connecting DP resources and making them communicate at a very simple level, bits, packets, strings, ...
- nor was it to pursue the dream of a universal business model such as the MIS dream of the seventies.

In between reconciling a view derived from an ideal CIM system - the CIM Target - and a top-down approach, predefined a priori scheme, a Reference Model. Such a vision which has driven the AMICE Consortium since the beginning is based on techniques well expressed in software engineering such as

- "objects" orientation
- instantiation process
- mapping of a model onto another.

The innovation which has justified the AMICE work is to bring those techniques into a coherent approach geared to the CIM users.

There are a number of issues, partially resolved at this stage which this paper will just mention but which the description of the architecture will put in light.

- genericity vs specificity

For many reasons genericity has economical values both for CIM providers and CIM users, economies of scale, quality of an individual CIM component reusable more often and therefore more reliable, or usable between different sites or even different companies.

However specificity is a fact of life. AMICE has never dreamed of imposing upon specific user major constraints for the sake of standardization, it is the other way around. We have to accept manufacturing business world as it is with its current constraints, with tremendous inertia and major differences in cycle times between various CIM components, hardware and software one cannot expect any virgin land, any steady state to start with. However instantiation techniques allow for reconciliation of both objectives.

- granularity

A CIM component can be small or large. The benefits of standardization hopefully to be derived from CIM-OSA architecture are not at all the same depending upon the "size" of a component.

Too large components will maintain distributed islands of proprietary closed subsystems limiting the benefits of the architecture.

Smaller grains will require a tougher consensus process and probably a multiplication of "adaptors" to transform existing components into standard components.

Although not so visible, the work already accomplished by AMICE towards the right granularity is of extreme value to the partners and eventually to the community.

4. CIM-OSA Models

Major aspects which underlie both the Enterprise and Implementation models are those of Function, Control, Information, Material and Resources. Material is separated out from Information because of the logistics associated with physical items.

The Enterprise Model describes "what" has to be done within the enterprise. It is also a partial description of "how" things are to be done since it expressed the flow of action within the enterprise. It thus describes the functional structure of a Particular Enterprise (the Function Architecture). It is concerned with the control structured (Rules) underlying the business process.

The Implementation Model is a description of the final physical realization (the Real World) of the total Enterprise as a system. It therefore describes machines, human resources, enterprise specific software, basic Data Processing resources and standard support services.

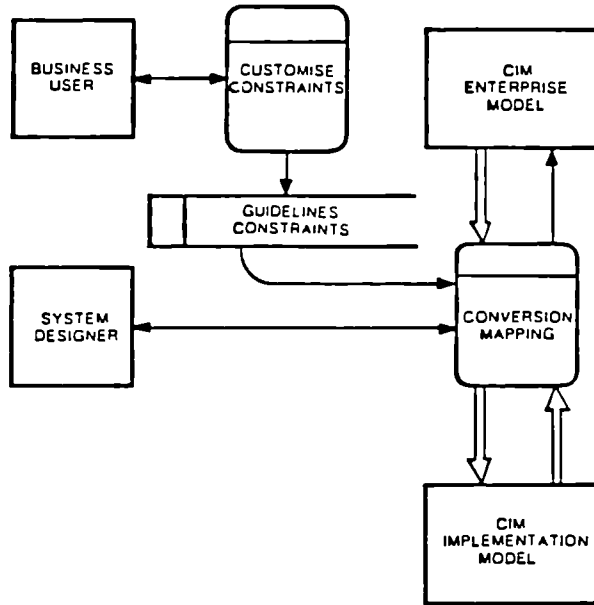


Figure 2 : CIM System Design

The Conversion Process provides the means for converting Enterprise Model into Implementation Models in a consistent manner. The process is controlled by the CIM-OSA Guidelines (see Figure 2) which can be customized by the Business User to reflect his own enterprise strategies. Thus it would be possible, for instance, to define the degree of interaction / autonomy and type of automation to be applied to any specific area of enterprise.

Ideally such a Conversion Process should be entirely automatic. However at the current level of knowledge it is only possible to create heuristic models. This means there can be no hard and fast rules for making implementation choices. It is therefore necessary to involve the services of a professional Systems Designer to oversee the conversion process. The Systems Designer will be asked to choose possible implementation choices at various stages in the Conversion Process.

The Conversion Process is basically one of regrouping the various aspects of, Function, Control, Information, Material, Resource in order to structure the activities and their organization into "cells" which can be implemented by CIM system components complying with CIM-OSA principles to provide a cost effective implementation that will meet the Particular Enterprise's strategic goals.

5. CIM-OSA SOLUTIONS

In order to control all stages in the Design Process for a CIM system it is necessary to define generic structures for the complete CIM enterprise as a system. At all stages of the Design Process the CIM-OSA Reference Architecture will provide the Basic Building Blocks and the Guidelines for their application. The main Building Blocks are described below in the context of the three levels of integration previously discussed.

The final result of CIM-OSA is a model of the real world implementation of CIM system - the Implementation Model - that truly reflects the specific enterprise requirements at an economic cost. The Implementation Model (Figure 3) is described in terms of the Implemented Components - those components chosen from Supplier's product Offerings which meet or exceed the minimum requirements derived from the Enterprise Model during the Conversion Process. The Implemented Components have to cover all aspects of Manufacturing and Information Technology i.e. machines, human resources, software specific to the enterprise, basic Data Processing resources and the CIM-OSA support services which collectively form the CIM-OSA Integrated Data Processing Environment.

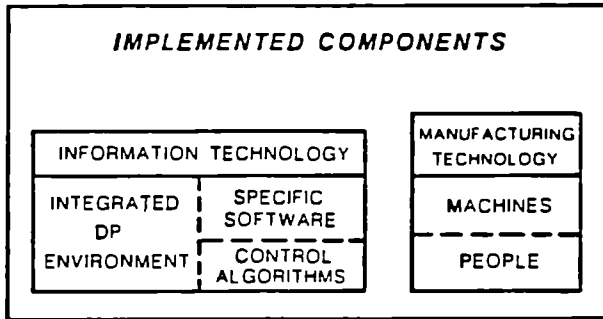


Figure 3: The Implementation Model

Within CIM-OSA the Integrated DP Environment (Figure 4) provides a range of services to support the concepts of Physical System, Application and Business Integration as developed above. These services ride on a set of Basic Data Processing Resources (implemented in a proprietary way) that provide full database access and management facilities, an OSI-compatible network system, and a

(possibly separate) set of functions for data communication with local (non-networked) system devices in addition to basic Data Processing services.

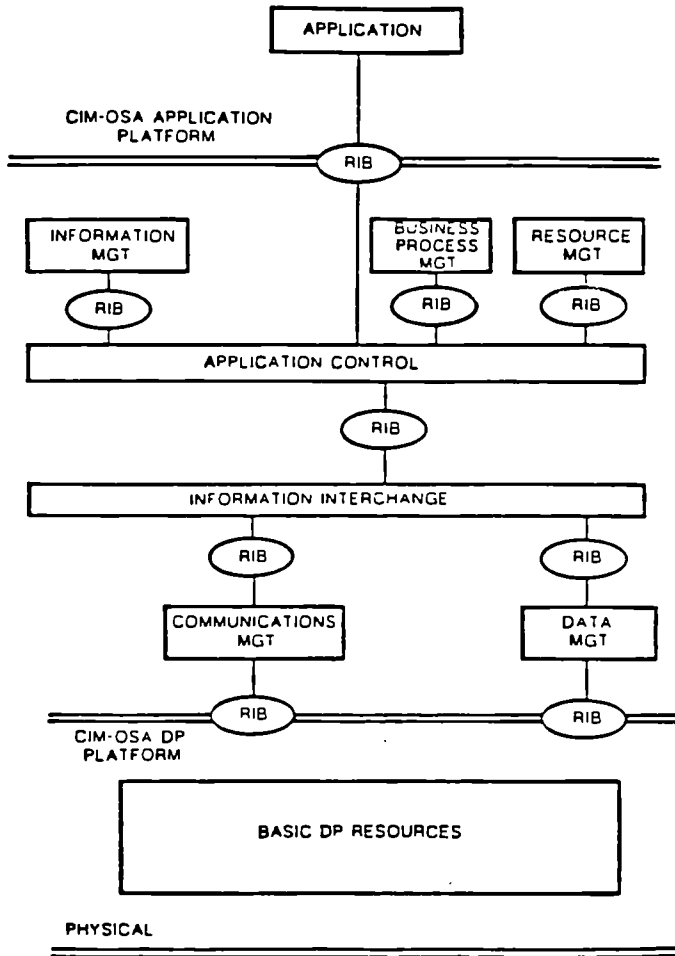


Figure 4: Integrated Data Processing Environment

Physical System Integration is the realm of inter system communication, network configuration and network management, data exchange rules and conventions, and physical system interconnection. It is concerned with the need to exchange information. Information Interchange provides a service to control the CIM system-wide exchange of information. This service will interact with peer counterparts, via the Communication Management service, in order to obtain information from another node, and via the Data Management to obtain information from the local database system. When communicating with external devices Information Interchange will also provide data conversions in accordance with the appropriate External Schema for the device. Communication Management provides a location-independent system-wide addressing facility whilst Data Management provides local address conversion to access the local database according to specific Internal Schemas.

It is envisaged that these services will be provided by OSI networked facilities in the Transport, Session, and Application Layers.

Application Integration is the realm of portable applications, distributed processing, common services / execution environments, and common shared data resources. It is concerned with the exchange of information between applications, and between applications and enterprise storage systems. In order to convert data into the representation required by the applications it will be necessary to perform system wide view conversions in accordance with the External Schema for the application. It will also be necessary to provide system-wide configuration control in order to maintain consistency between the many sets of information (view editions) existing within the enterprise. These services are provided by the Information Management Service which will be based on the OSI Presentation Level though it is foreseen that some CIM dependant enhancements may be needed.

It will also be necessary to provide an Application Control service to look after the run-time execution of Applications and provides them access to Information Management and the Physical Integration services. Applications are the processes which execute the functions defined in the model of the enterprise.

Business Process Integration is the realm of knowledge based decision support for business control, automated business process monitoring, and production and process simulation. It requires a Resource Management service to look after the maintenance and allocation of resources to ensure that the necessary resources are available when required for the execution of particular activities. A Business Process Management service is provided to control the sequencing and the synchronisation of activities by

invoking the rules specifying the flow of action. Business Process Management must have access to the Implementation Model for the current operational system so that it can find out where, when and how the processes or applications can be activated on the correct node(s).

We use an Enterprise Model in order to specify the total enterprise system requirements. Such requirements can be defined at several levels, the long term strategic view, the medium term tactical view and the short term operational view, the medium term tactical view and the short term operational view. Figure 5 shows the various Building Blocks used for Enterprise Modelling.

The basic constructs are the Business Process and the Enterprise Event. Business processes are groups of activities and the rules for interconnecting them to achieve a specific purpose. Enterprise Events trigger the execution of a Business Process. The Business Process is further decomposed into Business Rules and Enterprise Activities in order to describe the total requirements for the Enterprise system. There are two classes of rules, Declarative and Procedural. Procedural Rules define the connectivity (flow of action) required between Enterprise Activities at execution time, whilst Declarative Rules define the longer term strategic and tactical constraints under which the enterprise has to operate. The Enterprise Activity defines the required Function and the Inputs and the Outputs (Control, Information, Material, Resources) required to carry out a particular action in a business process. Information is subdivided into Product and Production. Product information describes the design of the actual products, whilst production information describes not only the characteristics of the production plant but also the information needed to control the day-to-day operation of the plant (eg. order details, production schedules, etc.).

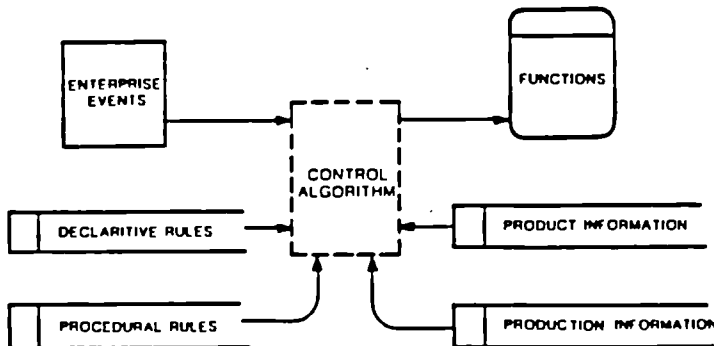


Figure 5: Enterprise Model Constructs

The Information description is completed by the creation of a Conceptual Schema which is the central rationalized description of all enterprise information.

Enterprise events (such as the receipt of a customer order) trigger the control algorithm to execute a specific sequence of activities. The control algorithm then selects the appropriate rule(s), requests, assignment of necessary resources, gets required information, passes control to and from the activity and checks status of the result.

Activities are selected from a small number of CIM-OSA Enterprise Building Blocks according to the relevant CIM-OSA Reference Model Guidelines. Such Building Blocks will be elementary in nature and as such will be common to several processes. Differentiation in specific actions being achieved by the Business Rules covering run-time parameterisation.

6. USE OF CIM-OSA

The process of using the CIM-OSA Building Blocks and Reference Model Guidelines to realize and enterprise specific CIM system is now outlined. The Business User is able to describe his Particular Enterprise at any state of the evolution he requires. He will be able to create, or modify, a set of consistent models to describe not only the current enterprise but also the long term strategic and medium term tactical aims. Having created a set of consistent models he is then able to compare them and develop strategies and plans for migrating from one to the other. These models will also require to be maintained in order to reflect changes in strategies required to align them with current achievements.

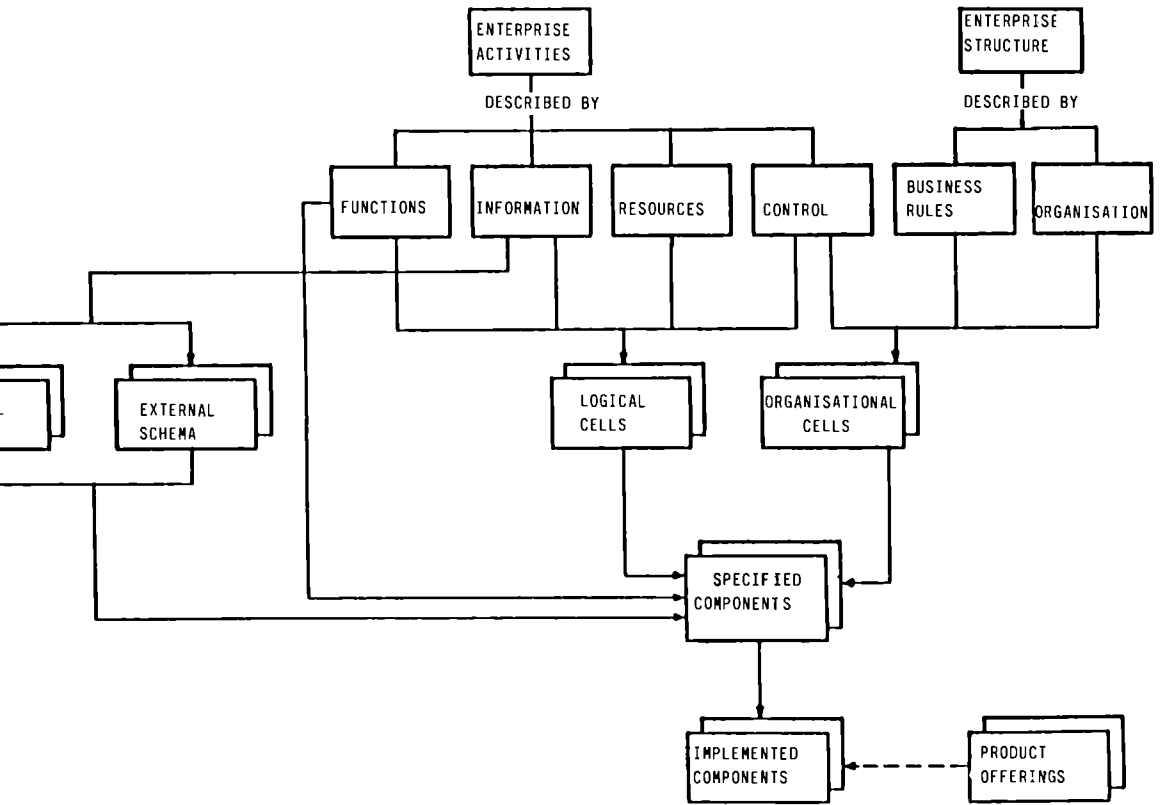
The model of the current situation will be used in executing the day-to-day operation of the Enterprise. Deviations from the anticipated/planned performance being reported to the Business User for immediate consideration and corrective action. Proposed corrective actions can then be evaluated on the model and the most successful solution implemented with delay.

Development of CIM system in accordance with CIM-OSA principles begins with the creation of the Particular Enterprise Model. The Business User creates and overall description of his Particular Enterprise, using the CIM-OSA concepts of Business Process, Enterprise Event, Business Rules and Enterprise Activity in order to capture the fundamental requirements. The Business user will be aided in this process by having available in a "knowledge base" sets of functional primitives (CIM-OSA Building

Blocks) which he can select from with the aid of an "interferencing engine" (CIM-OSA Guidelines).

The first step involved the selection of a number of Logical Cells and Organizational Cells. Logical Cells are constructs for the definition of a group of resources required to support a set of related Enterprise Activities (and their required information and resources). They may be determined for a given enterprise in a variety of ways. For instance, they may be used to reflect a job-oriented structure, or a process-oriented one. Their primary purpose is to identify groupings of functions which require close or frequent interaction and thus serve to highlight collections of equipment and resources which are candidates for having a high degree of integration. Organizational Cells are a construct to enable the assignment of responsibilities within an enterprise, providing a basis from which the timely provision of resources and information needed for the execution of activities can be enabled.

The Logical and Organizational Cells may be identified for a given enterprise by different selection criteria. Examples of these may be to minimize/simplify interactions between cells, or to maximize autonomy.



Now the Logical and Organization Cells are primarily concerned with the relationships between grouped activities, and do not fully reflect the detailed functions performed by these activities. Introducing specifications of the detailed functions themselves, results in the formation of Specified Components. These are therefore functional specifications of what is required to support the Logical/Organizational Cells.

The Conceptual Schema (Information description) contained in the Enterprise Model now has to be analyzed in order to determine the physical storage structure (enterprise filing system) by way of the Internal Schema. The methods of external access to the data must also be defined by External Schemas. The necessary Specified Components forming the Integrated Data Processing Environment part of the Implementation Model can then be defined. This completes the creation of an enterprise wide logical view (System Architecture) of the required CIM system.

Figure 6 for reasons of clarity, shows the Logical Cells and Organizational Cells being derived in an essentially separate way. It should be emphasized that in practice the analysis would be an iterative one where compromises are made to achieve the optimal identification of all cells.

The Implementation Model is now created by choosing, from actual Product Offerings, Implemented Components which meet or exceed the specifications of the Specified Components. The distinction between Specified and Implemented Components is made so that, for example, when the design characteristics of an Implemented Component exceeds the capabilities of the desired (Specified) component (such as a crane with 6 tonnes load capacity when only 4 tonnes capacity is required, or a 40Mbyte disk when only 20Mb storage are required) this information can be preserved for later use when expanding the system.

A further distinction between Specified and Implemented components is that each type of component is described only once as a Specified Component (e.g. a detailed specification of a required workstation, "X"), but may be replicated many times as an Implemented Component (e.g. "we have 20 workstations of type "X"). Each Implemented Component is therefore uniquely identified, and contains such additional information as the component supplier, serial number, location of use, etc.

Finally the physical implementation of the particular CIM system will be created by assembling the CIM-OSA compliant components specified in the Implementation Model.

the above systematic and straightforward approach to the design of CIM systems will probably not be wholly applicable to practical cases. Very often, implementation constraints, for example imposed by existing equipment, will be known from the very first step. Therefore, some Logical Cells may be defined before the Enterprise Activities they will support. CIM-OSA can cope with such situations, which may also be encountered while extending CIM systems, because the links existing at the Reference Levels between the elements of the various models may be used in both directions.

7. BENEFITS OF CIM-OSA

CIM-OSA will provide a well understood mechanism for defining the model of a specific enterprise and from that deriving a consistent design for the required CIM system.

CIM-OSA thus provides a frame of reference for CIM users to strategise, plan, implement, maintain, upgrade their enterprise system. An integrated methodology will prevent inconsistencies arising between the various levels of enterprise model.

Well understood and accepted standards based on CIM-OSA and adhered to by users and vendors will make available standard Information Technology Modules and Tools which can readily be fitted together to build a unique CIM system for a company (compatibility of hardware products and software products, portability of software products). This ease of integration will reduce overall costs and elapsed time scales so that medium and small companies will be able to justify the necessary expense of CIM and permit the introduction of changes more rapidly than the competition. This ease of integration also leads to flexibility of operation so that a rapid response to market changes can be made resulting in a more competitive enterprise. A reduction in manufacturing costs, development and production time, inventory lead time, order and delivery times will also result.

An open system will allow integration with foreign (non CIM-OSA) components and intercommunication with other systems.

CIM-OSA will define what has to be standardized in terms of Functions and their associated Interaction Boundaries. Components observing these interaction boundaries will be compatible with each other and therefore allow integration of the large number of very different subsystems required to realize a manufacturing enterprise at minimal cost. This ease of integration with existing systems providing a smooth stable transition from current state to a modified one.

CIM-OSA will promote the realization of CIM systems which are flexible in structure leading to supplier independence. The ability to integrate information across a company makes information available as and when required and in the exact form required by the end user. It will also be possible to design more autonomous enterprise structures in which decision making can be postponed until enough information is available. We call this decision postponement by constraint propagation using "least commitment principle". Thus each sub-system will be able to autonomously control its own region of responsibility by reacting to the events according to rules that encapsulate the experience and knowledge of the enterprise (decentralized decision making).

8. CONCLUSIONS

It has been shown that the trend toward standardization at higher and higher levels relates more and more to the semantic contents of the information used by the people at work in their various roles, in the case of CIM-OSA those of a manufacturing business.

The Consortium has been cautious so far not to demonstrate too early CIM in order to avoid any simplistic understanding of the aims of the project

It is now planned before the end of the project not only to deliver the architecture but to organize a representative demonstration on a real case.

Defining developing and validating a CIM architecture is a huge effort. The partners intend to pursue their effort and promote the architecture as it develops within the 19 businesses of the Consortium.

Selected applications will be proposed under ESPRIT II.

Project No. 1199

HUMAN SKILL AND COMPUTER POWER

Andrew Ainger
BICC Research and Engineering Ltd
38 Ariel Way, Wood Lane
LONDON W12 7DX

Dr Shaun Murphy
Greater London Enterprise Board
63-67 Newington Causeway
LONDON SE1 6BD

Recent developments in ESPRIT Project 1217 (1199) are discussed; in particular, a more refined definition of human-centred qualities of manufacturing technology, and the application of these ideas to the demonstration cell at BICC, one of the new partners in the project.

The object of the project is to produce a human-centred integrated CIM system, in which people are given responsibility for all tasks and perform those tasks best done using human skills, whilst the use of computerised systems in support of the shop floor operators is optimised.

One fundamental aspect of a human-centred CIM system is that shop-floor operators of a production cell, or production island, will have substantial responsibilities in contrast to conventional CIM.

One of the purposes of BICC's demonstration cell is to prove the cost effectiveness of the human-centred approach to CIM. The functional requirements specification has been drawn up, which has led to the identification of various interface issues. Since the human-centred CIM systems concept will have a considerable impact on the manufacturing infra-structure into which the cell is installed, the organisation structure of the company will have to be modified. The respective tasks of human operators and computers are discussed, the underlying manufacturing philosophy is described, and the required production planning information is outlined.

INTRODUCTION

The object of ESPRIT Project 1217 (1199) is to produce a human-centred integrated CIM system, in which people are given responsibility for those tasks best done using human skills, whilst the use of computerised systems in support of the shop floor operators is optimised.

Amongst the developments taking place within the project are: a cell or production island controller which provides the cell operator with computer support for scheduling jobs within the cell; a sophisticated lathe controller on which the lathe cell operator can produce part programs without using a programming language; and a drawing board on which traditional free hand sketching can take place and which transmits the drawing data to a CAD system.

Recent developments in the project are discussed in this paper, in particular a more refined definition of human-centred qualities of manufacturing technology, and the application of these ideas to the demonstration cell at BICC, one of the new partners in the project.

HUMAN CENTRED QUALITIES

Concern has been expressed by a number of authors in recent years about the lack of appreciation of the value of human skills and human judgement, and the consequences of this. (Refs 1,2,3 and 4).

Human-centred technology is that which emphasises human skill and responsibility. It is not possible to give a precise definition, but the following paragraphs describe aspects which are considered as fundamental:

1. Context

It is envisaged that manufacturing within the factory will be performed within production islands. Each production island will contain a number of machine tools and the appropriate number of operators. Full responsibility for production within each island will rest with the island operators.

2. Retention and enhancement of existing skills

A human-centred system is one in which traditional design and manufacturing skills are used to their fullest extent. Whilst a system may be used by people with varying degrees of skills, it will be most productive when used by skilled people.

The nature of skills will change with time, but at present, for example, the skills appropriate for a turning cell are those of a turner who has learned the trade on traditional lathes, and who has some experience of NC machines.

3. Maximisation of operator choice and control

The person should control the machine rather than the machine controlling the person. Choice and control should be optimised in a number of respects. In particular, the timing of actions and events should be decided by the operator. The operator should be able to move freely around the shop floor, and outside it if necessary. People should be able to communicate freely with each other, both formally and informally, and both electronically and face to face. Within the framework of production targets, the operators should be personally responsible for organising their work in such a way as to achieve those targets. Choice and personal control in these areas will automatically regulate the amount of stress that people are subjected to.

4. Subdivision of the Work should be Minimised

The range of activities for which each operator is responsible should be maximised. In this sense human assertiveness is the antithesis of Tayloristic scientific management. The aim of Taylorism is to sub-divide work as finally as possible in order that each minute element can be prescribed exactly and performed by someone with as little intelligence and training as possible and who does not have to exhibit initiative of any kind.

This aspect of human assertiveness implies a high degree of skill and qualification gained through both experience and formal education and training.

5. Operator Knowledge of the whole Production Process should be Maximised

Each person should know about the whole production process, and should have the opportunity to comment on any aspect of it. The forms of human-centred systems should be such that knowledge of the whole production process is easily acquired, and communication between people working at different points in the system should be facilitated. Communication should be both formal and informal. For example it may be that communication between designers and turners may be more productive if it is a conversation over a drawing rather than around an electronic screen.

6. Physical Ergonomic Factors should be Optimised

These ergonomic factors include, for example, the physical arrangement of the cell, design of keyboards, and user friendly aspects of software. These factors as well as, for example, safety considerations are very important. A system could hardly be described as human-centred if it was not engineered adequately; however, these factors do not in themselves constitute fundamentally human-centred qualities.

THE BENEFITS

In brief, the benefits of human-centred CIM are thought likely to be twofold.

1. Human-centred technology will provide more dignity and more fulfilling jobs for workers in manufacturing industry. It is, therefore, culturally desirable, especially in the context of the European tradition of craft-based manufacture.

2. Combining those tasks which skilled workers can best perform with those tasks that computers can do well, will produce a powerful human-centred system which will be more flexible than an automated CIM. Particularly for small-batch production, it is likely to be more efficient, more productive and, therefore, more competitive than conventional systems.

PROJECT OBJECTIVES

The purpose of the project, is to prove the flexibility and robustness of the human-centred approach by demonstrating machine/cell programming, planning, scheduling, and design, in a human-centred environment.

The primary objectives of the project are the following:

- a) Establish criteria for the design of human-centred systems
- b) Establish their economic and commercial competitiveness
- c) Achieve a higher level of flexibility and robustness against unforeseen events than is possible with conventional systems.
- d) Achieve a lower throughput time from design to finished components than is possible in conventional automated systems
- e) Achieve a better working environment and working practices by developing the synergy arising from man-machine interaction
- f) Assess and further develop the effectiveness of man-machine interaction
- g) Establish the means by which the training of operators can be most successfully carried out
- h) Design flexibility into the modules, to facilitate portability and re-configuration for other systems
- i) By the success of the demonstration, show that there is a better means of organising production, which is specially suited to European conditions.

This will be done by means of publication, demonstrations, communication with other CIM work, seminars and other means.

A list of the project partners is shown in the appendix. The German partners are carrying out the computer-aided planning (CAP) work; the British are doing the computer-aided manufacturing developments (CAM); and the Danes are performing the computer-aided design work (CAD).

PROJECT PROGRESS

A paper on criteria for human-centred systems was presented at the ESPRIT Technical Week last year (Ref5).

A recurring theme over the past year has been the meaning of human-centredness, and the social scientists have described this approach at length (Ref6). The paper discusses the three principal methods which can be used; the use of work dimensions and human-machine interface design criteria, the use of a design scenario, and the participation of users in the design process.

The Danish partners have established a number of CAD user groups, and have received a substantial amount of material from these groups which is influencing the design of the electronic drawing board.

The principal design decisions which define the layout of the human-centred turning-cell, as well as the lathe control and programming software, have been taken. The CAP philosophy has been agreed and decisions have been taken on those planning levels on which most development work will take place.

The new partners BICC and Rolls-Royce will demonstrate manufacturing cells embodying human-centred technology developed by the project, and tailored to their own industrial needs. The BICC work is discussed in some detail later in the paper. There is also discussion about a fully integrated CIM demonstration system, including, CAP CAM and CAD at the BITZ on the campus of the University of Bremen.

A DEMONSTRATION-SITE PERSPECTIVE

For more than a hundred years the philosophy of automation for batch manufacturing industries has been orientated towards cost reduction through de-manning or de-skilling of manufacturing tasks. Industry has been moving inexorably towards an unmanned factory of the future concept. This concept is abhorrent to many people and has been actively campaigned against by organised labour groups since the British Industrial Revolution.

There is now, however, a growing realisation that the human being still possesses several qualities which can be utilised to enhance the capabilities of a Computer Integrated Manufacturing (CIM) system, and it is for this reason that the concept of a human-centred CIM system is now being proposed. Some of these qualities could be recreated by current state-of-the-art computing techniques, but only at a cost which is entirely inappropriate to the manufacturing process, particularly in small to medium sized businesses.

A human being is not well equipped to carry out tasks which require the following:

- constant vigilance
- repetitive work
- consistency
- multi-variable monitoring etc

These tasks can, however, be readily computerised.

The qualities which a human being can contribute are as follows:

- The ability to deal with unforeseen events
- Intuition
- The ability to handle complex quality criteria eg: texture, visual attributes etc.
- The ability to operate at a higher rate than normal for short periods to provide recovery potential
- The ability to change tasks rapidly with minimum delay etc.

These qualities can only be automated with a significant cost and complexity penalty with the latter tending to degrade reliability. As direct labour costs are frequently not the greatest element of a product's cost, it is often more beneficial to attempt to reduce inventory and stock levels (and the consequent holding costs) by utilising labour more effectively to manufacture products more rapidly, than to attempt to reduce labour costs.

In recent years, attempts to achieve these manufacturing improvements have tended towards the development of manufacturing strategies which are often systems-centred in their approach to manufacturing control ie: centrally operated computer systems dictate the activities of the shop floor. Manufacturing Resource Planning (MRPII) is an example of this and it is becoming increasingly obvious that this does not represent a panacea for European industry. For example, some time ago, a certain factory was sending across the shop floor jobs which were scheduled by the factory planner, to be made on a particular machine. When this schedule arrived at the shop floor, the supervisor promptly replanned his entire work load, ignoring the planner detailed instructions. When asked why he did this, he indicated that the particular machine the planner had loaded the jobs onto had been scrapped two years ago. An alternative to this system-centred strategy is the human-centred approach to manufacturing where people, assisted by computer systems, make the decisions which affect their own activities.

UNDERLYING MANUFACTURING PHILOSOPHY

The concept of human-centred computer integrated manufacturing could be applied to several different types of manufacturing philosophy. It is however, the belief of a growing number of people that the greatest benefits will accrue from applying computer techniques within an organisation which is directed towards human-centred manufacture.

Consequently, the following has been adopted as the underlying manufacturing philosophy for this particular demonstration site.

Manufacture of, ideally products, or at least complete sub-assemblies, will be carried out in small quasi-autonomous units (cells). Each cell will be responsible for all the activities within it, including quality, due-date performance, first line maintenance etc. Each cell will be staffed by a team. Individual team members may be expected to embrace more than one function. It should also be recognised that different individuals will possess different intellectual levels such that some may respond positively to job enhancements in terms of task rotation and increased responsibility whilst others may be quite content to maintain their existing work patterns.

PRODUCTION STRATEGY

Several businesses have identified the need to reduce manufacturing lead times such that production can be planned against customer orders rather than sales forecast. To achieve this, however, is likely to require a gradual reduction in production batch sizes which will increase the complexities for any planning system attempting to control the entire factory.

The use of a cell-based manufacturing organisation, however, offers the advantage of simplifying the planning requirements by sub-dividing them into two levels, factory level and cell level.

Factory Level:

Factory customer orders will be planned, using Optimised Production Technology (OPT)/ Just-in-Time (JIT) techniques. Orders will be offered to the manufacturing cells for discreet planning periods (say 1 week), rather than on a quasi-continuous basis, using the principle of Period Batch Control (Ref7). As a result of the interactive planning process, a final list of orders will be agreed between the cell manager and the factory planner. This could then be used by a Material Requirement Planning (MRPI) system to determine all the relevant works orders and/or procurement orders.

The factory planner will receive, from each cell, information concerning the quantity of the product produced plus any relevant data concerned with: the products quality (or lack of it), the products late arrival/delivery to the next cell, aggregated cell performance figures for that product etc.

Cell level:

Within each cell the list of jobs offered by the factory planner will be sequenced and scheduled for the current production planning period to achieve a minimum manufacturing time based upon local knowledge of resource availability and/or specific constraints within the cell. The cell manager/team will be able to assess all data connected with the manufacture of the product, such as product quality details, plant health information and process data etc.

It is the intention to use period batch control utilising single cycle, equal length time periods and only assemble in the current period products which can be delivered to customers in the next period; only manufacture in the current period parts which are required for assembly in the next period: only procure raw material in the current period for manufacturing the parts in the next period which are required for assembly in the period after next. See Figure 1.

PRINCIPLES OF PERIOD BATCH CONTROL

Diagrammatic representation of the flow of orders
using a repeated single-cycle system

Time period Cycle number	1	2	3	4	5	6	7	8	9	10	11	12
1	P	R	M	A	D							
2		P	R	M	A	D						
3			P	R	M	A	D					
4				P	R	M	A	D				
5					P	R	M	A	D			

Key

- P = sales order processing and planning phase
- R = raw material and /or component call-off phase
- M = manufacturing phase
- A = assembly phase
- D = despatch phase

Example

Products which are scheduled to be manufactured in period 4 were ordered in period 2 and will be deliverable in period 6

Fig. 1

The number of jobs scheduled for each time period for each cell must be based upon the principle that 'everything that the cell is required to carry out for those jobs must be achievable within the current time period.' In this way each cell is theoretically de-coupled in time, from its preceding and succeeding cells. However, it is anticipated that in the event of a transient manufacturing disturbance, such as an unplanned resource breakdown, which might prevent a cell from completing all its jobs within the time period, a local agreement could be arranged between the cell manager and the manager of the succeeding cell to ensure that the effects of the disturbance are minimised by positioning the affected job(s) at the end of the sequence in the succeeding cell. This facility is open to abuse and must, therefore, be carefully controlled to prevent this type of interaction from becoming a regular occurrence.

Once the cell managers have determined whether their cells are able to achieve the initial schedule, the factory planner must attempt to re-organise those products which cannot be produced by use of such techniques as:

- offering the jobs to other cells capable of manufacturing the product (albeit at lower efficiency)
- offering jobs earlier to the cells with future predictable overloads
- sub-contracting

The result of this interactive liaison between factory planner and cell manager should be an achievable work schedule which effectively forms an agreed contract of work for the cell for the period of time under consideration.

Future Work Areas

The effectiveness of the solutions proposed by the 1217(1199) programme of work will be analysed in terms of their impact upon business performance (cost reduction, manufacturing lead-time reduction etc) and in terms of their sociological impact based upon criteria established by the social science working group (Ref6).

APPENDIX

The following Partners are Members of the Consortium:

Greater London Enterprise Board (Prime Contractor)
 UMIST, Manchester
 Rolls Royce, Leavesden
 BICC, London
 RD Projects (Sub-Contractor), London
 University of Bremen
 Krupp Atlas Elektronik, Bremen
 Technical University of Denmark
 Technical Institute, Copenhagen
 NEH Consulting Engineers, Copenhagen

REFERENCES

1. Rosenbrock H. The Future of Control, Proc 6th IFAC Congress, Boston (1976)
Reprint in Automatica Vol 14 (1977)pp 389-392
2. Rosenbrock H. The Redirection of Technology. Proc IFAC Conf Bari (1979)
3. Cooley M. Architect or Bee. The Human Price of Technology. The Hogarth Press. London 1987
4. Weizenbaum J. Computer Power and Human Reason. Penguin Books. London 1984.
5. Crampton S. Design, Engineering, Management and Control. ESPRIT Technical Week. September 1986.
6. Rauner F. Rasmussen L. Corbett M. The Social Shaping of Technology and Work. To be published in Artificial Intelligence and Society.
7. Burbridge J.L. Period Batch Control, CIT, 1985

VI. INFORMATION EXCHANGE SYSTEMS

Parallel Session

- 1. Network Infrastructure: Development Projects 1807**

Project No. 718

The OSI-PC – An Introduction

Title: **The OSI-PC
– An Introduction**

Author: **Dann Holst Sommer,
Systems Programmer**

Project: **CARLOS,
Esprit proj. ref. no. 718**

Country: **Denmark**

Company: **RC Computer**

Abstract

The OSI-PC constitutes one of the components being developed by the CARLOS project. The objective of the CARLOS project is to provide the users of public data networks with value added services according to ISO standards for open systems interconnection (OSI).

The OSI-PC is a personal computer (PC) communicating across an X.25 service in compliance with the OSI protocols.

When using the OSI-PC the user obtains the following novelties:

1. Independence of vendor when interconnecting heterogenous systems
2. Integration of the PC into the OSI-concept

This paper gives an introductory presentation of the functionality and the services of the OSI-PC.

The OSI-PC, AN OVERVIEW

The OSI-PC is a personal computer (PC) communicating across an X.25 service in compliance with the OSI protocols.

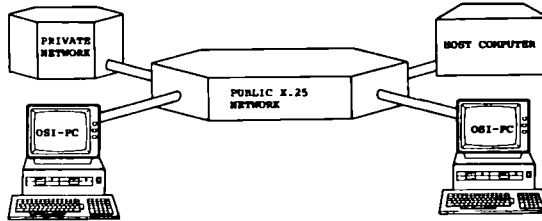


Fig. 1. The OSI-PC, System Context

The OSI-PC provides the basis for the following services:

- file transfer
- electronic mail
- host access

None of these services are new, however. The novelties of the OSI-PC are:

1. Independence of vendor when interconnecting heterogeneous systems
2. Integration of multiple functions within a single component

The former is possible due to the fact that communication is performed in conformance with international standards. I.e. the user at the OSI-PC may access multiple applications/users irrespective of the vendor of the equipment in the other end.

The latter novelty is provided through the integration of the above functions into a PC, providing a versatile communications workstation. Thus the user is given the freedom of opting for local data/processing, i.e. using the PC directly, or remote data/processing, i.e. access through the network. To put it another way: in a distributed environment the user may choose the level of distribution which will be optimum for the specific environment.

The presumed user of the OSI-PC will be utilizing non-trivial systems, typically in multi-user configurations, which demand a microcomputer supporting substantial processing requirements.

State of the Art

Mainframe implementations reaching the upper layers of the OSI-model have been introduced by major European computer manufacturers.

Commercially available implementations of the uppermost layers are still part of the future. It is expected, however, that such implementations will be available in only a few years. This assumption being based upon the activity which may be registered within the research environment.

Adapter boards for microcomputers as well as microcomputers providing the X.25 level 1-3 service are at present available from several manufacturers. Adapter boards providing services above level 3 across a WAN are as yet unknown, making the OSI-PC an obvious candidate for setting the trend of OSI-communications for microcomputers.

Implementation of the layers 1-7 for a PC is currently taking place within the scope of CARLOS - the CARLOS OSI-PC.

With the OSI-PC a solid platform is established facilitating the implementation of OSI-based applications for a wide range of microcomputers.

Configurations

The OSI-PC supports several different configurations, namely:

- the lower layer PC (LOLA)
- the middle layer PC (MILA)
- the upper layer PC (ULA)
- the all-layer PC (ALLA)

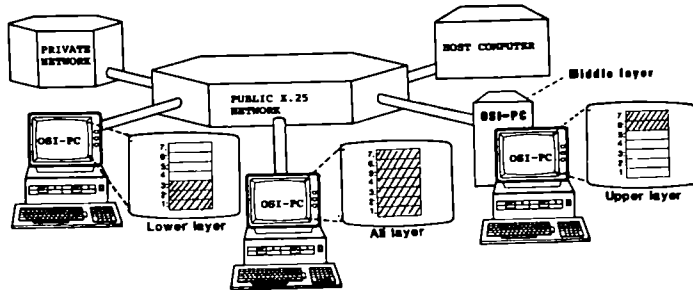


Fig. 2. OSI-PC Configurations

The lower layer OSI-PC provides a level 3 service, i.e. transmission of messages on virtual connections, implementing multiple connections on a single transmission line.

The middle layer PC provides a level 5 service, i.e. data exchanged between application processes through connections, establishing and terminating connections, and sending end-to-end messages and controller dialogues.

The upper layer PC represents a minimum dedicated solution providing access via a LAN. The X.25 network is accessed through a network component comprising layers 1-5, e.g. a middle layer PC.

Both the all-layer and the upper layer OSI-PC facilitate a level 7 service, i.e. end-to-end service with independence of representation of data.

This service may either be provided by the all-layer PC with layers 1-7 or by the upper layer PC with layers 6-7 only.

The all-layer PC represents a general standardized component providing direct access to the X.25 network.

Services of the OSI-PC

The services which may be facilitated by the OSI-PC are:

- Network service (X.25 DTE)
- Triple X service (An integrated PAD)
- Session layer service (A LAN server)
- Virtual Terminal (VT)
- File Transfer, Access and Management (FTAM)
- Electronic Mail (X.400)

- thus providing a very versatile communications component for the OSI concept.

Sample Applications

The utilization of the OSI-PC by the user is illustrated through two representative examples with emphasis on the services of the upper layers.

Virtual Terminal (VT)

The VT is used for host access. The OSI-PC may access multiple hosts irrespective of the vendor of the actual host computer as long as the host application performs the terminal access in accordance with the VT-protocol.

Within the OSI-PC a mapping will be performed to/from the local terminal and the VT-format.

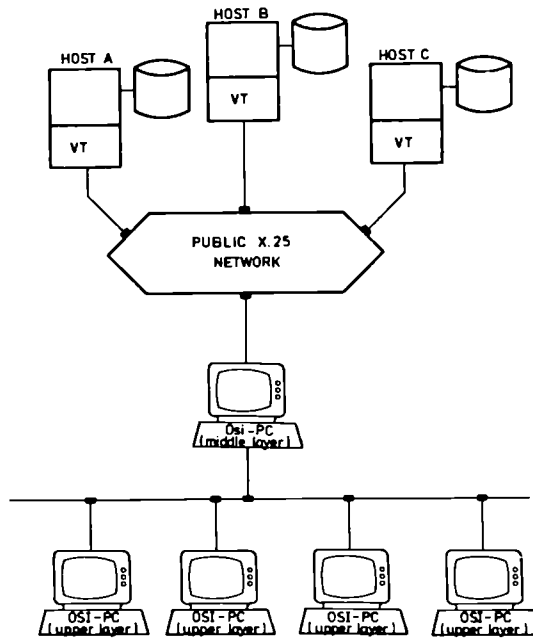


Fig. 3. Virtual Terminal, Sample Configuration

The sample configuration illustrates upper layer OSI-PCs accessing multiple hosts via a middle layer OSI-PC, each of them utilizing otherwise incompatible concepts for interworking with terminals.

File Transfer, Access and Management (FTAM)

FTAM is used for the exchange of files.

The OSI-PC may exchange files with any computer with a storage device as long as the file transfer is performed in accordance with the FTAM-protocol.

Both within the OSI-PC and the host computer a mapping will be performed to/from the local file system.

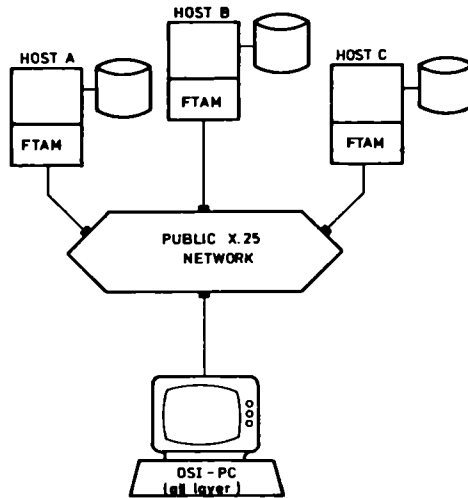


Fig. 4. FTAM, Sample Configuration

The sample configuration illustrates an all layer OSI-PC accessing multiple hosts, each of them utilizing otherwise incongruous concepts for information storage.

Standards Compliance

The present implementation of the OSI-PC will comply with the following standards:

Layer, Standard	Comments										
VT, DIS 9041	The implementation supports the s-mode of operations. The s-mode default profile is supported. The subsets VT-A and VT-B are implemented.										
FTAM, DIS 8571	The implementation acts as an initiator <table border="0"> <tr> <td><u>Option:</u></td> <td><u>Supported:</u></td> </tr> <tr> <td>file service class:</td> <td>file transfer</td> </tr> <tr> <td>functional units:</td> <td>kernel, read, write, grouping, limited file management</td> </tr> <tr> <td>service level:</td> <td>UCFS</td> </tr> <tr> <td>attributes:</td> <td>kernel</td> </tr> </table> <u>Attribute value range:</u> filename: max. 1 component up to 64 chars content type: MAR document type: unstructured binary file constraint set: unstructured abstract syntax: simple binary	<u>Option:</u>	<u>Supported:</u>	file service class:	file transfer	functional units:	kernel, read, write, grouping, limited file management	service level:	UCFS	attributes:	kernel
<u>Option:</u>	<u>Supported:</u>										
file service class:	file transfer										
functional units:	kernel, read, write, grouping, limited file management										
service level:	UCFS										
attributes:	kernel										
CASE, DIS 8649											
PL, DIS 8823	The functional unit kernel and context management are implemented										
SL, IS 8327	The following functional units are implemented: <ul style="list-style-type: none"> - kernel - half-duplex - typed data - capability data exchange - minor synchronize - resynchronize - exceptions - activity management 										
TL, IS 8073	Class 0										
X.25/3, X.25, 84											
X.25/2, LAPB, 84											
X.25/1, X21 bis											

Fig. 5. Standards utilized

Present Status

The implementation of the OSI-PC will be completed at the end of 1987.

Implementation of the lower three layers has been completed at present.

Implementation of layers 4 and 5 is completed late summer 1987 and the implementation of layers 6 and 7 at the end of 1987.

Future Plans

Development of the OSI-PC concept will continue at two fronts:

1. Developing new services for the OSI-PC
2. Alignment of the OSI-PC with the de facto microcomputer standards of today

This work will be carried through in the SESTA project, the implementation of the OSI-PC on a separate adapter board, this project being a spin-off from CARLOS. Furthermore, the possibilities of re-using X.400 SW from the CACTUS project for the SESTA project are currently being considered.

Further Information

Apply to:

RC Computer A/S
19, Klamsagervej
DK-8230 Aabyhoej
Denmark

Phone: + 45 6 25 04 11
Telex: 64 169 rcarh dk
Telefax: + 45 6 25 09 92

References

1. "ESPRIT Project, Sub-Area 6 IES, CARLOS Technical Annex", 87.04.01
RC Computer

Project No. 718

CACTUS, An X400 System for Private Management Domains

Author: R. D. Killick M.A., CASE Communications Limited

CACTUS (CARLOS Addition for Clustered Terminal User Agents) is a project within the European Community's ESPRIT programme which implements the X400 Recommendations of the CCITT in a manner intended for medium sized organisations. In the terms of the standards, the CACTUS device is a shared-resource user agent for clusters of terminals. It enables the large existing population of IBM and compatible Personal Computers to be utilised thus providing a migration path to X400 for these users and devices.

CACTUS Consortium

As with all ESPRIT projects, part funding comes from the European Community and the rest from the Consortium Partners. The CACTUS consortium consists of:

CASE Communications Ltd - A major UK-based manufacturer of data transmission, data networking and message handling systems for world markets.

Fischer and Lorenz - A Danish communications consultancy who specialise in OSI work and project management.

University of Madrid - Working on OSI since the late seventies including the areas of Transport Layer and protocol development tools.

University of Catalonia (Barcelona) - Have been involved in work related to making OSI software available on small computers.

CACTUS Functionality

CACTUS allows groups of people using personal computers to access X400 facilities transported over X25 networks. These facilities may be public or private.

The CACTUS deliverable consists of software which can support up to 100 mailboxes and thus approximately 100 users. The users gain access to the device by means of one of at least four ports running asynchronously at 2400bps. The X25 connection operates at 9600bps.

The CACTUS design allows for future connection to simple terminal and host computers. It also permits future implementations expanding on the parameters given above.

The figure shows both the environment and the software structure of CACTUS.

A number of personal computers are connected to the CACTUS box. This connection can be a direct connection or over a dial up telephone network. The architecture permits future implementations using Local Area Network connection.

The CACTUS box is connected to an X25 data network and, thence, to other X400 devices or services. These can be public (administration management domains) or private.

The user operates a personal computer to prepare messages. In order that the user of a CACTUS does not have to learn the intricacies of yet another word processing package or editor for message preparation, the user's favourite word processing package can run concurrently with the PC portion of the CACTUS software system. This gives the user complete freedom to select a package whilst providing all the message preparation facilities required.

Following message preparation, the user invokes the CACTUS code within the PC and can then send to the CACTUS Box for delivery to the intended recipients when communications to the destinations can be established via X25. (This allows for line congestion and deliberately delayed messages which take advantage of lower line tariff rates at certain times of day.)

Incoming messages are delivered to a CACTUS mailbox based on information in the message "envelope". The user logs into the mailbox at intervals and receives notification of any messages waiting. The user can then decide how best to dispose of the message, e.g. read, reply, store or discard.

The CACTUS device itself is managed from a PC and a wide range of Manager facilities are available.

CACTUS Software

The X400 defined protocols P1 (for inter-MTA communications) and P2 (for inter-User Agent communications) are implemented. The mailbox system exists partly in the CACTUS device and partly in the PC. The two parts of the mailbox use the P7 protocol to communicate and the Remote Operations Server (ROS) to ensure error-free data transfer between them.

The Mailbox Client/Server pair and the Remote Operations Server (ROS) are implemented in conformance with the European Computer Manufacturers Association proposed standard P7. It is expected that this standard will be substantially ratified in 1988 as part of the 1988 X400 recommendations. By intercepting this standard, CACTUS will be one of the first realisations of this valuable functionality.

The link between PC and CACTUS Box makes use of a Simplified Reliable Transfer Service. This simplification is possible because the PC, when in use, is directly connected to the Box. The design of the Service draws upon the experience obtained in the CARLOS project. Architecturally, it could be replaced very readily, by the full RTS if, for example, the PCs were separated from the Box by a LAN.

The component marked Session Server consists of the lower layers of the OSI tower. This includes the Basic Activity Subset of Session layer, Transport Classes 0, 2 and 4 and both 1980 and 1984 implementations of X25. It will be ported from the CARLOS project, in which CASE Communications is also a partner.

All the remaining components in the diagram are being produced specifically within the project.

The Message Transfer Agent (MTA) is the X400 concept of a message router. In this implementation it acts as an end system and does not perform onward routing, or relaying, of messages.

The specifications of all the modules defined by standards within the Box are being written in the formal description language ESTELLE, making use of the ASN.1 compiler.

To facilitate testing an Operational Data Recorder module has been added to the MTA.

A list of the X400 service elements supported by CACTUS is given in the appendix.

To make a complete implementation, many aspects not covered by standards have had to be considered. The three most significant of these, the Man Machine Interface, Local Management and Addressing are discussed further below.

CACTUS Hardware

The CACTUS hardware vehicle is expected to be VME-bus based and will support the UNIX operating system for the higher layers of software on one processor board and a simpler operating kernel on another processor board for the Session Server.

The PC is expected to be an IBM PC or compatible supporting MS-DOS.

User-Machine Interface

This part of the design of the CACTUS system has received much consideration as the presentation and ease-of-use of the system play a major part in user acceptance.

It has been decided that a single interface will not satisfy all users. Beginners like menu systems that guide them carefully with prompts and help at every step but more experienced users become annoyed by menus because of the perceived "messaging about" to move around the menu tree to reach the next desired command. Conversely an interface designed specifically for an expert user with little or no command or parameter prompting with type-ahead enabled causes confusion and panic in the novice. Therefore a dual interface is provided which can support both modes of operation. There is also a degree of adaptability of the interface by which, if a user logged on as an expert does not seem to be getting very far, the interface reverts to a menu system.

Management

The human supervisor who has to manage the CACTUS system communicates with the system from a PC just like any other user. He has access, however, to a number of mailboxes called Task mailboxes, which have special characteristics.

The content of messages sent to Task mailboxes can be processed by specially written tasks. These implement commands which the supervisor may need to give, for example, to permit a new user to come onto the system.

Users, as well as the supervisor, need to communicate with the manager software. Special HELP and MANAGER mailboxes are provided. These can receive messages from any user and are regularly examined by the manager software.

The manager has a separate command set which allows the supervisor to exercise control in areas such as:

- User maintenance (add, delete, modify profile)
- Mailbox maintenance
- System startup and shutdown
- Broadcast message to all users
- Network control and test
- Directory maintenance
- Statistics (Traffic, Queues, Problems....)
- Accounting
- Backup and archiving

Addressing

Users frequently send messages to the same address. To avoid entering the full O/R name each time a message is sent the user can set up a directory entry for that address indexed by a friendly reference. Messages can then be addressed using this shorter reference instead of the full address.

A user has a user-maintainable directory in the PC. The CACTUS box holds a central directory maintainable only by the supervisor through the manager software.

CACTUS support a central directory and PC held directories.

PC directories are maintained by the user.

A user can request maintenance of the central directory through requests (messages) to the supervisor.

Each Message Domain (MD), public or private, will define its addressing scheme according to its particular needs. This means that in order to reach a recipient inside an MD, the necessary Originator/Recipient (O/R) Name fields will depend on each particular MD. The format of O/R names is defined by standards.

Besides the above, CACTUS has defined an internal addressing scheme for routing messages which uses a subset of the fields within the full O/R name.

The addressing scheme inside CACTUS takes into consideration the following O/R Name fields:

- Country Name
- Administration Domain Name
- Private Domain Name
- Organization Name
- Organizational Unit

The identification of a message varies according to the level of operation. Within a PC the user may identify a message by, say, moving a cursor down an index of message subjects. The PC would then interrogate CACTUS using a more global identification.

The Interpersonal Message ID is the only unique identifier over the whole X400 world. It is attached to an outgoing message by CACTUS. Incoming messages have the IPmessageID attached by their sending device.

Conclusion

CACTUS will provide a cost effective solution to the problem of accessing the X400 world intermittently from personal computers.

The architecture allows the development of gateways or servers which can provide X400 access for groups of users other than through the conventional large mainframes.

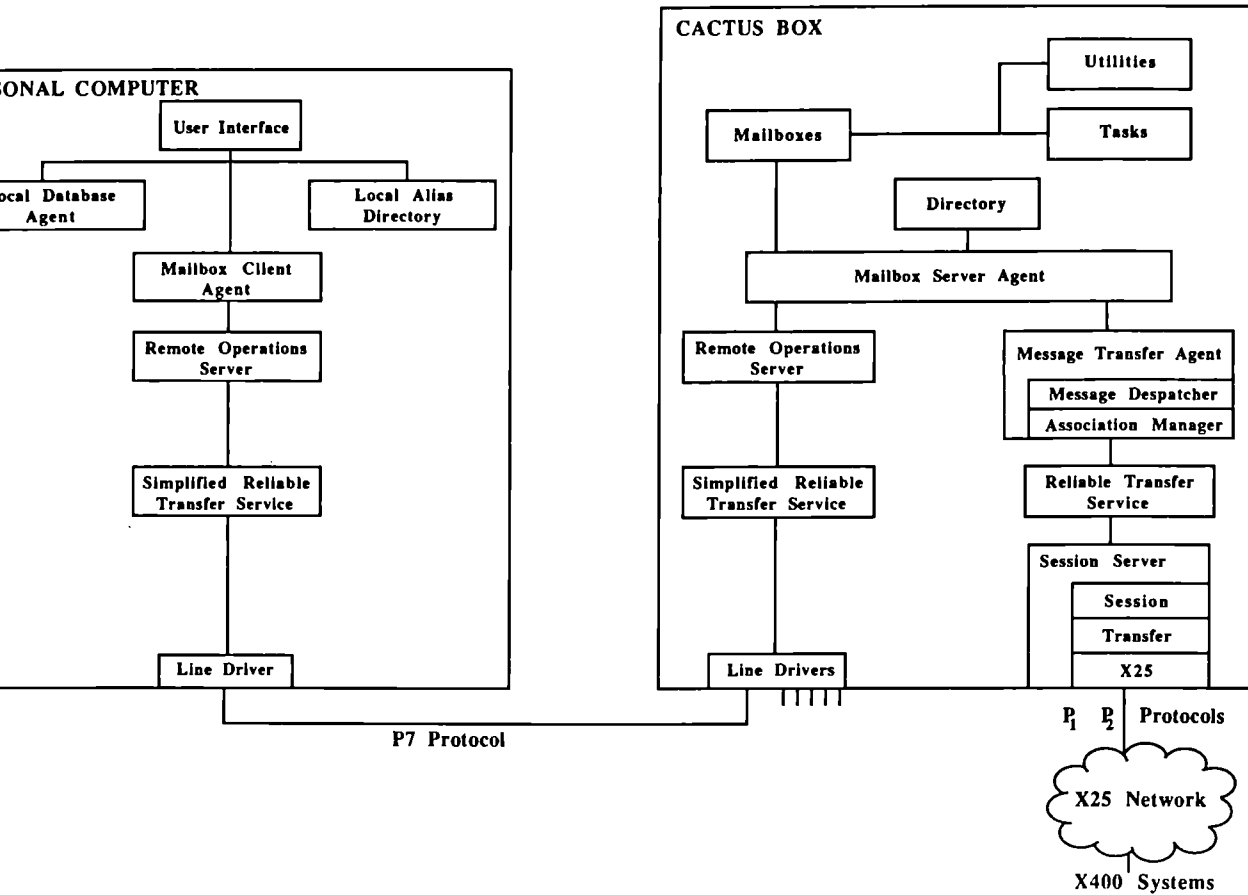
Appendix

X400 Service Elements Supported by CACTUS

The support by CACTUS of the listed service elements is given for origination or reception. A simple YES means supported for both origination and reception a NO means not supported at all. This is a set of facilities for the first version and may be extended in the future.

Alternate Recipient Allowed	YES
Authorising Users Indication	YES reception
Auto-forward Indication	YES reception
Blind Copy Recipient Indication	YES reception
Body Part Encryption Indication	YES reception
Conversion Prohibition	YES
Cross-referencing Indication	YES
Deferred Delivery	YES origination
Deferred Delivery Cancellation	NO
Delivery Notification	YES origination
Disclosure of Other Recipients	YES reception
Expiry Date Indication	YES reception
Explicit Conversion	NO
Forwarded IP-Message Indication	YES reception
Grade of Delivery Selection	YES
Hold for Delivery	NO
Implicit Conversion	NO
Importance Indication	YES
Multi-destination Delivery	YES origination
Multi-part Body	YES
Non-Receipt Notification	YES
Obsoleting Indication	YES
Originator Indication	YES
Prevention of Non-Delivery Notification	NO
Primary and Copy Recipients Indication	YES
Probe	NO
Receipt Notification	YES
Reply Request Indication	YES
Replying IP-message Indication	YES
Return of Contents	YES
Sensitivity Indication	YES
Subject Indication	YES

FUNCTIONAL OVERVIEW OF CACTUS



IMPLEMENTATION OF A NETWORK ADMINISTRATION
CONFORM TO THE OSI ARCHITECTURE IN THE
ESPRIT INFORMATION EXCHANGE SYSTEM

W. BLUMANN (SIEMENS) - J. COLASSE (BULL) - T. COOK (GEC)
C. REMY (BULL) - G. THOMAS (ICL) - B. WINTER (SIEMENS)

ABSTRACT

The ROSE project (Research Open Systems for Europe) is a major development project of the ESPRIT Information Exchange System (IES).

The aim of the project is to develop prototype OSI software for UNIX* machines through a phased approach, the current phase being the third one.

The tasks fall into two categories :

- tasks consisting of upgrading existing services (X400 mail, FTAM) to ISO standards, or CCITT standards, with alignment to CEN/CENELEC or SPAG/GUS profiles.
- tasks relative to services entirely developed during the third phase (Remote operation services, Network management).

The main development of the third phase is the Network Administration task. It is based on a unique specification defined by all partners, and must be conform to the OSI management architecture. It will take into account the three basic components proposed by ISO :

- SMP (System Management Process)
- HIB (Management Information Base)
- An administrative channel.

* UNIX is a registered trademark of AT&T Bell Laboratories.

1. INTRODUCTION

In this section we summarise the objectives of the ROSE project. Later sections discuss in more detail the remote operation service and the main development of the third phase : the network administration.

1.1 Overall aim of ROSE

The ROSE project (Research Open Systems for Europe) is one of many projects launched under the ESPRIT programme by the Commission of the European Communities in 1984. It is a major development project funded through the IES budget. The project has brought together five major European computer manufacturers (BULL, GEC, ICL, Olivetti, and Siemens), and a number of smaller companies and research organisations in the development of software for Open Systems Interconnection (OSI).

The major features of the ROSE software are :

- provision of the communication and network management services required by European collaborative research in general and the ESPRIT research community in particular
- conformance to ISO and CCITT standards
- use of the UNIX as the preferred operating system
- availability for a range of different makes and types of computer hardware, software environments and communication networks

1.2 Overview of the project

The ROSE project is essentially a software development and integration project. Its aim is to develop prototype OSI software for UNIX machines marketed by the partners. It implements an operational network providing the following services between the contractors :

- remote login and job execution
- electronic mail and conferencing
- file transfer for document exchange and software dissemination
- access to and handling of remote files

These services are provided between heterogeneous machines and terminals with a maximum use of public networks and services. The implementations of these services are based on the relevant ISO and CCITT standards with alignment, where possible, to SPAG or CEN/CENELEC profiles.

Associated with these services there will be a network management service and facilities for management of a distributed directory of users. The distributed directory service is the responsibility of a parallel ESPRIT project called THORN (The Obviously Required Name Server).

UNIX System V has been chosen as the preferred operating system for ROSE because it has the following advantages :

- many European manufacturers provide it for their products and are committed to it for future developments
- it is widely used by the international research community
- it offers a good range of development tools for the production of portable software

ROSE software provides a transport service conforming to ISO DIS 8072/8073. Classes 0, 2 and 3 are provided over the public X.25 networks and class 4 over CSMA/CD local area networks.

UNIX provides UNIX to UNIX file copy (uucp) and mail facilities for use over the telephone network. These facilities are widely used by the UNIX user community. The ROSE project has adapted them to run over the ROSE ISO transport service to offer :

- a migration path for existing UNIX sites that wish to move to OSI
- a way of providing gateways between the existing UNIX network and the ISO network provided by ROSE

Two standard OSI applications are provided by the ROSE software :

- a message handling system (MHS) conforming to the CCITT X400 series of standards
- a file transfer service (FTAM) conforming to the ISO DIS 8571 standard

The ROSE software already provides a full implementation of the ISO session protocol so that these and other presentation and application layer software can be built to run over it.

1.3 Work in the third phase

The main directions of the final phase of the ROSE project are :

- Continuation of the pilot installation phase to increase the level of testing of the ROSE software
- Harmonization of proprietary OSI implementations
- Reinforcing interconnectivity by allowing the interconnection of IANs (each considered as a subnetwork) using the connectionless Internet protocol
- Enhancing the applications (X.400, FTAM) already provided in Year 1 in order to bring them into alignment with the appropriate international profiles
- Provision of a Remote Operation Service (ROS)
- Development of network administration software allowing the basic management of a heterogeneous ROSE network.

2. REMOTE OPERATION SERVICE

The design of the remote operation service considers some basic guidelines :

- * The boundary between the ROS-applications and the RO-service has to be modelled by an explicit programming interface including association control.
- * The interface has to be independent of the semantics of the ROS-applications.
- * The communicating applications must be transparent with respect to the configuration of the communicating processes.
- * No migration to more advanced standard versions emerging during the development period of the product shall take place.

Following these basic guidelines the major design decisions are explained in more detail in the next subclauses.

2.1 Standardisation documents

The standardisation documents valid for ROS-V1 are :

- * CCITT - Draft X.ros0 = working document for DIS 9072/1

Remote Operations :
Model, Notation and Service Definition
(Version 4, Munich, Feb. 1987)

- * CCITT - Draft X.ros1 = Working document for DIS 9072/2

Remote Operations :
Protocol Specification
(version 4, Munich, Feb. 1987)

- * ISO/SC21-N881-DIS8649/2

Service Definition for Common Application
Service Elements
Part 2 : Association Control, 86-06-12

- * ISO/SC21-N882-DIS8650/2

Protocol Specification for Common Application
Service Elements
Part 2 : Association Control, 86-06-12

- * 2nd ISO/DIS8824

Specification of Abstract Syntax
Notation One (ASN.1), 86-05-28

- * 2nd ISO/DIS8825

Specification of Basic Encoding Rules for Abstract Syntax
Notation one (ASN.1), 86-05-28

* ISO/DIS8822

Connection Oriented Presentation Service Definition, 86-06-12

* ISO/DIS8823

Connection Oriented Presentation Protocol Specification, 86-06-12

* ISO 8326/27

Basic Connection Oriented Session Service Definition and Protocol Specification
(no unlimited user data during connection establishment)

2.2 Private profile definitions

2.2.1 Remote operations protocol

Arguments have to be encoded with explicit length description.

All association- and operation classes are supported. Linked operations are not supported.

Only the mapping on presentation is provided. But it should be noted that this mapping is in principle applicable to all applications. Whereas the alternative mapping to the reliable transfer service is detrimental for applications like system management.

The requirements for presentation functional units are : Presentation kernel functional unit.

The requirements for session functional units are : kernel functional unit, duplex functional unit.

2.2.2 Association control protocol

Only mandatory parameters are supported. The background for this is the limited amount of user data currently negotiable during the session connection establishment phase.

2.2.3 Presentation protocol

Only mandatory parameters are supported.

There is a one to one relationship between a presentation address and the corresponding transport address in both directions.

2.3 Interface to the ROS-service

The highlights of the interface are different techniques of data and control flow between the service user and the service provider and the configuration independence.

From the point of view of the control flow the following alternatives apply :

- * synchronous or asynchronous confirmed services

Synchronous means :

A request service primitive and the corresponding confirmation service primitive are combined to a "request confirmation" function call.

Asynchronous means :

Request and confirmation service primitives correspond to distinct function calls.

* Concatenated or non concatenated function calls :

Concatenated means :

One "super" functions call can include for example a set of distinct invoke requests.

Non concatenated means :

A function call is dedicated to exactly one instantiation of a specific service primitive.

From the point of view of the data flow the following alternatives apply :

* fragmented or non fragmented data

Fragmented means :

the fragmented data portions are allocated to distinct and non adjacent containers.

non fragmented means :

data has to be allocated completely to one data container

* data in buffers or in files

From the point of view of the configuration independence the following statements hold :

* Prior to calling the ROS-interface the application process has to translate application entity titles to presentation addresses.

* The ROS-application and the ROS-interface are independent of the real location of the RO-service.

* The provider of the RO-service can be located in one system, in two strongly coupled systems interconnected without a network and in two loosely coupled systems interconnected via an OSI-network.

The outlined interface has been submitted to XOPEN.

The actual interface implementation within ROSE has some restrictions which touch in no way the global OSI-connectivity :

* There is only an OSI-service provider.

* Only asynchronous confirmed services are supported.

* A function call corresponds to exactly one service primitive.

* Only non fragmented data (arguments) can be surrendered.

2.4 Protocol machines

The ROS-interface including the AC-interface maps directly on the session-interface. From the abstract point of view that leads to strongly coupled AC-, RO-, and P-PM. From a more concrete point of view there is no distinct P-PM visible (in terms of external interfaces). For the association management the PPM is included in the ACPM and for the transfer management the PPM is included in ROPM. But the distinction between encoding/decoding routines for AC-PDU, ROSE-PDU and PPDU is strictly maintained.

3. NETWORK ADMINISTRATION

3.1 Introduction

In addition to the production, transportation, and processing of data within computer networks, there are requirements for functions that will ensure the correct operation of the entire system and the interoperation of all its components. These functions, which range from planning, system startup and shutdown, monitoring and control to error diagnosis, performance measurement and accounting are referred to as management functions.

The cooperation of systems of different manufacturers in an OSI environment requires efficient system management tools on basis of international standards. Contrary to popular belief management standards are emerging and progressing well. The rapidly growing activities on standardization work in the system management area were encouraging to start work on implementing network management in a real OSI environment based on these emerging standards. The ROSE network seemed to be an excellent platform for such an enterprise.

The project, which started in October 1986, is part of the year 2 of the ESPRIT ROSE project and is called Basic Network Administration. The elected project title expresses that it is not the primary goal of this project to provide for comprehensive and powerful system management products covering all functional areas. Limited budget and manpower impose restrictions in the selection of the functions to be provided. Therefore the project will mainly concentrate on proving the validity of the OSI Management architecture as defined in the ISO/ECMA Management Framework papers and on the realization of management functions to access and present statistical information extracted from the transport and session layer. These statistical information will fit particularly well into the performance management area.

The intention of the project is to follow as close as possible the ongoing standardization work. Still existing gaps in the standardization work have been filled, as far as the ROSE Basic Network Administration project is concerned, by the so called Blue Profile on systems management prepared by SPAG companies. Except for use in the ROSE project this SPAG Blue Profile is intended for wide promotion as a strawman to assist the development of European and international management standards. Therefore another important goal for this project will be to gain practical experience with the application of the SPAG recommendations in a real OSI environment and to produce feedback for current and outstanding standardization work.

3.2 Architectural background

In a general definition, management encompasses all the activities needed to initiate, monitor, terminate, and control individual computer systems as well

as complex computer networks. The management activities relate to the information transfer and information processing resources of these computer systems. Those resources are called managed objects.

Two sets of management activities can be defined :

- activities related to the management needs of individual systems
- activities related to the interworking of management entities across connected systems

Systems which complies with the requirements of OSI standards in its communication with other systems are called open systems. Some aspects of the management of real systems are part some other aspects are not part of open systems. Only those aspects which are part of open systems (OSI environment) are relevant to OSI management.

The OSI Management Framework as described in ISO and ECMA documentation provides for the concept and abstract model for the identification of areas for developing management standards. The basic principles of the concept and the model are described below.

OSI Management is accomplished by :

- systems management
- (N)-layer management
- (N)-layer operation

Systems management provides mechanisms for the monitoring, control and coordination of all managed objects within open systems. Management may be modelled as being affected through a set of management processes. All these processes are not necessarily located at one local system but may be distributed in many ways over a number of systems. Where management processes in different systems need to communicate with one another they communicate using OSI Management protocols.

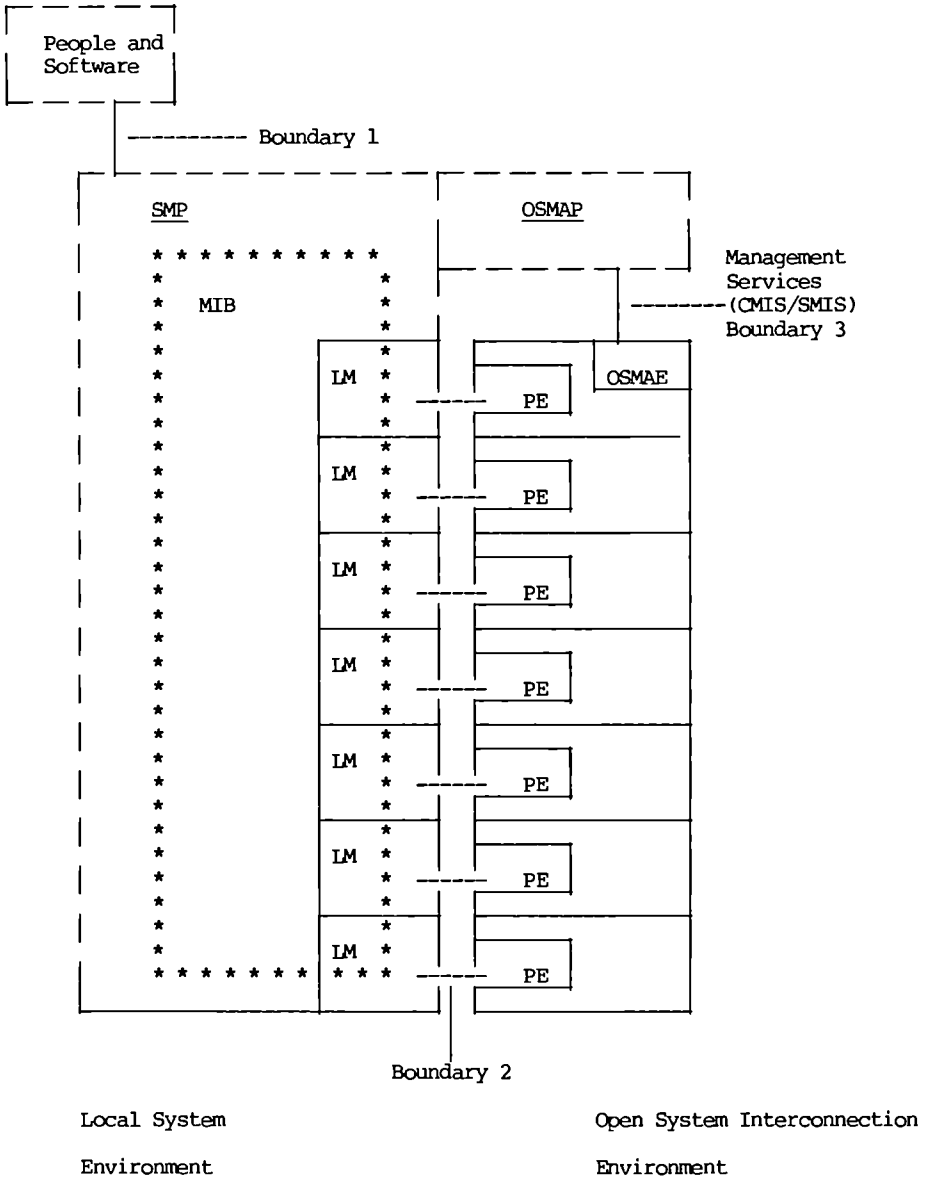
According to the ECMA Management Framework (see diagram 1) systems management of one open system is performed by the Systems Management Process (SMP). When communication of management information to or from SMP's in other open systems is required, it is realized by the Open System Management Application Process (OSMAP) component of the SMP utilizing the services of the Open System Management Application Entity (OSMAE). Each OSMAP will utilize the services according to its role in the overall systems management.

(N)-layer management provides mechanisms for the monitoring, control, and coordination of those managed objects used to accomplish communication activities within an (N)-layer.

According to the OSI principle that each layer is independent of all others, each layer may have its own layer manager (LM). Layer management is the collection of these (n)-LM's.

(N)-layer operation provides the mechanisms for the monitoring and control of a single instance of communication.

Diagram 1 : OSI Management Architecture (ECMA)



The management information base (MIB) is a conceptual composite of information relevant for an individual system to participate in the management of the OSI environment.

From a conceptual point of view the Management Information Base (MIB) and the Layer Managers (LM) can be considered as part of the SMP.

The Management model encompasses a boundary between the SMP and the people and software that request services of the SMP. Service requests/responses may pass through this boundary to invoke one or more systems management functions. This boundary is within the local environment and not subject to OSI Management standards.

ISO SC21/WG4 is actually working on a multi-part standard which defines Management Information Services (MIS). One set of services is intended to provide common services that apply to all aspects of management (Common Management Information Service CMIS).

Further sets of services will provide services for the specific functional areas of management (e.g. configuration, fault, performance management etc.). These service elements are called Specific Management Information Services (SMIS). The exchange of management information between the OSMAE's of different open systems will be supported by the Common Management Information Protocol (CMIP) and the Specific Management Information Protocol (SMIP).

3.3 Concept of the ROSE management project

The concept of the ROSE management project has been modelled according to the OSI management architecture. Diagram 2 depicts the general systems management structure which is applicable for each of the ROSE systems.

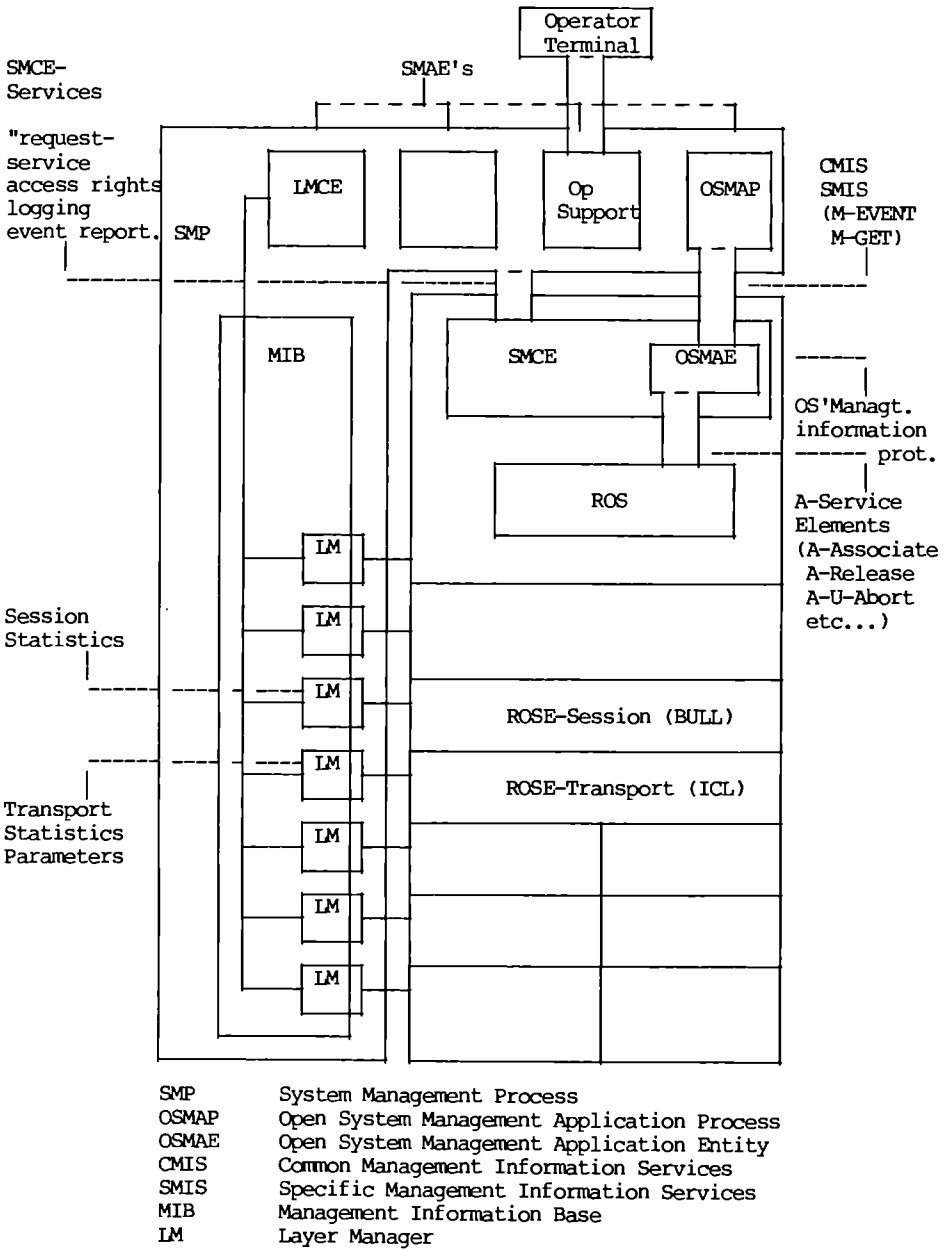
The main elements of the ROSE management architecture are :

- Systems Management Process SMP
- System Management Control Entity SMCE
- Layer Managers LM
- Management Information Base MIB

The SMP consists of a set of components which all together provide the functions to manage the local system and/or to take part in the management of other remotely connected ROSE systems. These components are referred to as System Management Application Entities (SMAE). SMAE's may have the following functions :

- communication with the Layer Managers (performed by the LMCE Layer Management Control Entity)
 - provides for specific management facilities, e.g. for performance management
 - provides for an interface between the systems administrator and the SMP (operator support)
- etc.

Diagram 2 : Concept of the ROSE Management Project



In order to manage ROSE systems remotely, management information has to be communicated between the SMP's of different systems. To perform this communication an OSMAP has been added to the SMP. From an architectural point of view the OSMAP can also be considered as an SMAE.

An SMAE can only communicate with another SMAE in the same system through the use of services provided by the SMCE. A so called "Request"-service enables an SMAE to request a management activity to be performed in another SMAE and to return results or requested information to the requestor (local "Request"-service). In addition to the delivery of such requests and results the SMCE provides for access control and logging functions in order to guarantee that only authorized processes (persons) may utilize the provided management services.

The OSMAE can be considered as being a part of the SMCE. A second set of SMCE services will be provided by the OSMAE. These services are a subset of the standardized CMIS services and can be used by the OSMAP in order to transfer management information to an SMP in another ROSE system.

A specific service is provided by the OSMAE for the delivery of a "Request" for performing a management activity in an SMAE in a remote ROSE system (remote "Request" -service).

The Layer Managers control the normal operation of the layers and deliver management relevant information on request to the SMP. The layer managers are represented in the SMP by an SMAE called layer management control entity (LMCE). All management activities to be performed by anyone of the layer managers must be addressed to the LMCE of the relevant system.

The MIB is the recipient of all management relevant information of an individual system. The form of physical or logical storage and the access to the management information is realized according implementation specific aspects of the ROSE systems on which the MIB is located.

3.4 Applied standards

As stated above the ROSE management project follows as close as possible the standards defined until now within international standardization organizations. In some areas where no standards are available at the moment the SPAG Blue Profile will be applied.

However, some areas of the management project are not yet considered by any standardization work. This is in particular true for functions like access control, logging, event filtering and distribution, providing an interface to a human administrator, etc. Therefore, in the ROSE Management project the realization of these functions are based on vendor specific principles. Due to this fact, practical experience in a real management system running in an OSI environment could be very important for influencing the outstanding standardization work in these management areas.

As far as applicable, the following standards have been applied for the ROSE project :

1. ISO/TC97/SC21 Information Processing Systems
OSI Reference Model - Part 4 : Management Framework

2. ECMA TR/37 Framework for OSI Management
3. ISO/TC97/SC21 Information Processing Systems
Management Information Service Definition - Part 2 :
Common Management Information Service Definition
4. ISO/TC97/SC21 Information Processing systems
Management Information Protocol Definition - Part 2 :
Common Management Information Protocol Definition
5. ISO DIS 9072/1
Information Processing Systems - Text Communication
Remote Operations / Part 1 :
Model, Notation and Service Definition
6. ISO DIS 9072/2
Information Processing Systems - Text Communication
Remote Operations / Part 2 :
Protocol Specification
7. Guide to the Use of Standards Part D4 :
Administration in an OSI Framework
8. Guide to the Use of Standards
zz Private Systems Management (Blue Profile)

The following chapters gives an overview about used subsets of the standards and restrictions or modifications, necessary for different reasons :

Common Management Information Services (CMIS)

From CMIS the services M-INITIALIZE, M-TERMINATE, M-ABORT, M-GET, M-EVENT REPORT will be used.

In order to support specific management functions of the ROSE system, e.g. access control, logging etc., a "private" service element has been defined : M-PRIVATE. This private service is not yet standardized.

Use of Remote Operation Services (ROS)

The following table contains an overview how the CMIS elements are mapped on-to CO-ROS elements :

M-GET request	----->	INVOKE request
M-PRIVATE request	----->	INVOKE request
M-EVENT request	----->	INVOKE request
M-GET confirme	----->	RESULT request
M-PRIVATE confirme	----->	RESULT request

Use of Association Control Service (ACS)

It is assumed that the association necessary for the information exchange between two management entities in different systems (SMCE's) will be permanently established. Once an association is established (invoked by the first management activity after system initialization) this association remains established and can be used for all following information exchanges until a system closedown occurs.

3.5 Functions

The described concept of the ROSE management project allows in a flexible way to provide for management functions in an OSI environment composed of systems from different vendors using UNIX or UNIX compatible operating systems.

One essential objective of this project is to demonstrate for such an environment the applicability of mechanisms developed in order to access locally or remotely to management information and to present the information in a suitable and user friendly form to the network administrator.

To access management information in remote systems the necessary services and communication facilities for the information exchange will be provided. Additional facilities allow the control of the access rights of a human administrator or administrator process, and the logging of all management relevant activities in a logging file. An event reporting facility together with a filter mechanism will be applied.

The management functions provided by this project are concentrated on the functional area relevant to performance management. However, the applied principles for gathering locally or remotely management information from OSI layers in order to present statistics to an administrator is applicable to all other management functional areas.

The following paragraphs give, in more detail, an overview about the most important functions realized within the ROSE Management Project.

Performance Management

The necessary statistical information for performance management is derived from the session and transport layer. The collection of these statistics will be done by the Layer Managers by means of statistical counters. For accessing these statistics, e.g. by the Performance Management SMAE, the Layer Management Control Entity LMCE must be addressed.

The following list gives an overview about the statistics provided for the transport layer. The choice of these counters has been done according proposals in the SPAG Blue Profile.

- User data transmitted
- User data received
- Exp. user data transmitted
- Exp. user data received
- Transmitted TPDUs
- Retransmitted TPDUs
- Received TPDUs

NO. DATA TPDU retransmissions
 Number of open connections
 Inbound successful connections
 Outbound successful connections
 Unsuccessful inbound connections
 Unsuccessful outbound connections
 Timed out connections
 Protocol errors
 Number of invalid TPDUs

Control of Access Rights

The management components in the ROSE management project provides for facilities to assign access rights to administrators or administrator processes in order to restrict the access to management functions to authorized persons only.

The scope of functions one administrator is allowed to invoke is defined within a so called profile. Each delivered command is checked by the SMCE in order to control whether the requestor is allowed to invoke this command or not (according the assigned profile for the requestor).

For easier use of profiles sets of commands can be comprised in command classes. Commands are provided for modification of command classes and profiles.

The identification (authentication) of an administrator is performed by means of a password.

Logging

All management activities, e.g. commands, results, event reports etc. can be logged in logging files. A time stamp will be added to each entry. A set of logging files is provided in order to prevent overwriting of logging information in case of system shutdown and restart.

Commands will be provided to manage the use of logging files and to edit logging information.

Event Reporting

A mechanisms is provided in order to distribute events. Events can be delivered to local or remote SMAE's. A filter mechanism is introduced, enabling the event receiving SMAE's to define a specific event or a set of events that should be delivered to these SMAEs.

Operator Support

The Operator Support provides for the interface between a human administrator and the system management components of the ROSE project. To evoke management activities in the local or in remote systems a set of management commands has been defined for utilizing by the administrator. Examples of administrator commands are :

DUSAT	define user authorization
DPRAT	define system authorization
CPROF	change profile
RAI	request administration information (for event filtering)
INFRA	informe about RAI commands

After a syntax check by the operation support the command will be forwarded to the SMCE component for delivery to the relevant SMAE for execution. In the opposite direction result information prepared by the command executing SMAE will be received from the the SMCE and delivered to the administrator.

3.6 Realization aspects

The ROSE Management Project had been started in October 1986. Target date for finishing the project is the first quarter of 1988.

The project makes good progresses. At the moment most of the specifications have been prepared. For the implementation of the components of the project the following steps have been defined :

- Realization of a stable "local" system comprising SMCE, Access Control, Logging, Event Reporting, Operator support
- Add Performance Management SMAE and LMCE to the "local" system (Bull)
- Add Session Layer supplemented with statistic counters (Bull)
- Add Transport Layer supplemented with statistic counters (ICL)
- Add OSMAE/OSMAP to the "local" system and test interworking (Siemens)
- Port the complete system to other partner systems and test interworking (Siemens, Bull, ICL, GEC).

Project No. 719

THE ESPRIT DIRECTORY SYSTEM

Franco SIROVICH (*)

S.W. sas
Via Torino 241
I-10015 Ivrea, Italy

The paper describes *THORN* (an acronym for *THE Obviously Required Name Server*) which is a Directory System designed for the ESPRIT Information Exchange System. The aims and the functionalities *THORN* provides to application programs are discussed, and related to the Directory System standards being defined by ISO and CCITT.

1. INTRODUCTION

The ESPRIT Task Force is supporting the development of a European-wide Information Exchange System (IES) is encouraging the adoption of the Open System Interconnection (OSI) communication model and related standards. The ESPRIT IES will involve a large number of *Communication Entities (CE)*, including humans, host computers, peripherals, and computer application processes, such as electronic mail and file transfer handlers.

When the scale of the communication network increases (and the ESPRIT IES will certainly provide a significant test-bed as far as scale is concerned), a number of problems become apparent which are all related to CE naming. Although CE naming schemes have been defined for all communication protocol layers, these schemes do not address the following problems:

1. The location stability problem. In a complex communication network, CEs often change their (physical) location in the network, either because the communication network is modified or because the location of the CE needs to be changed. Thus, the location of a CE should always be transparent to the accessing CE.
2. The CE location problem. A CE which is willing to communicate with another CE must then be able to use a description (let us call it a *name*) to locate the desired CE instead of an address, whatever communication protocol layer is used. There is thus the need of being able to map descriptions onto OSI addresses (possibly at all levels).
3. The user friendliness problem. Ultimately, names will be visible (and used by) human users (be they end users or system managers). The naming schema adopted by the various OSI protocol layers, including the user oriented ones such as X.400, can hardly be considered as user friendly.

Since a long time [1], it has been recognized that the above problems can fully be addressed only by introducing in the communication network a *Directory Service (DS)* to be used by all

* tel. +39.125.231712, uucp mail address ..lmcvax@iconet@sirovich.

The author is also Computer Science Professor at the Department of Computer Science, University of Torino, Corso Svizzera 185, I-10100.

CEs. The standardization bodies also are taking care of such a requirement, and an intense activity is going on, from the initial work in ECMA [2] to the current joint activity of ISO and CCITT [3]. The type of service being envisaged by the standardization bodies is quite more complex than the one offered by early implementations of DSs, in order to cope with the functionality and scalability required by the OSI services of the next years. The establishment of an OSI standard on DSs is of paramount importance since it should be clear that the utility of a directory service in many applications crucially depends on having globally agreed standards.

The THORN project [4] aims at implementing a DS that on one side will be useful to the management of many applications, and on the other side will be as close as possible to the emerging standards.

2. PROJECT OVERVIEW

THORN was launched at the beginning of 1985 as a three year project. We are now in the middle of the third year since some delay has been accumulated in the first year. A description of the progress and results of THORN at the end of Year 1 can be found in [4].

2.1. The Project Objectives

The overall objective of THORN is to build a DS adequate for the European Information Exchange Service, whose development is supported by the Council of European Communities under the aegis of ESPRIT. The objective of THORN can be characterized as follows:

1. To specify and build a *distributed* directory service which is in line with the emerging international standards.
2. To implement the *Directory Service Agents (DSAs)* and the *Directory User Agents (DUAs)* for the above service on a wide number of different computers.
3. To integrate the above services with a number of pilot application services operated by the participating organizations, for the purposes of at least the participants and possibly of larger experimental communities.
4. To provide early feedback on the adequacy of the provided level of directory services, and on the feasibility and performance aspects of such level of services.

From the above objectives, the project organization has been derived as follows:

1. Specification and implementation of THORN, exploiting experience and know-how about the emerging international standards and user community requirements.
2. Rehosting of the THORN implementation on the systems of the participating organizations.
3. Running of (large scale) pilot exercise(s) to provide the required feedback on both functional specifications and implementation choices.

The THORN consortium composition is directly derived from the project objectives and organization.

2.2. The THORN Partners

The THORN partners can be considered as belonging to two categories: *industrial partners* and *research partners*. The industrial partners are Olivetti, Bull, GEC, ICL, Siemens and S.W. (an Italian software house). Olivetti run the project and S.W. are providing the implementation effort. The other industrial partners are porting the S.W. implementation on their computer systems so that it can be demonstrated in a heterogeneous environment.

The research partners are CERN (the European Laboratory for Nuclear Physics), CNUCE (a research centre of the Italian National Research Council which runs a large computer network), DFN (Deutsche Forschung Netz, the German Academic Research Network), and UCL (Department of Computer Science, University College of London).

The research partners and the industrial partners together carry on the functional specification effort, while the role of the research partner is crucial in conveying to the project the user community requirements, and the feedback they can acquire from running pilot exercise which exploit the THORN services.

A close contact with the on-going standardization efforts is provided by some of the industrial and research partners, whose participating personnel also attend the standardization meetings as company or national representatives.

2.3. The Project Time-scale

Since the required project span (three years) is quite long, and the standardization environment is evolving quite fast, the project has been scheduled around three main phases, which provide software releases. Thus, implementation can proceed in parallel with changes in international draft standards, experimentation with a previous implementation and feedback to a next implementation. This staged approach leads to an evolving design, which tries to capitalize on previous releases in order to decrease the development costs and time.

The project plan provides three successive versions of THORN:

Year 1: A centralized DSA has been developed on an Olivetti machine. The DUA, developed on an Olivetti machine has been ported on several partners machines, including some of the research partners' ones. The protocol between DUA and DSA has followed the ECMA proposal [2] (no Aliasing nor Yellow Pages Services). A limited scale pilot exercise has been run using Year 1 software.

Year 2: The Directory Service has been decentralized and enhanced: replication and navigation has been introduced. The status of the project at the end of Year 2 will be described in the following. A large scale pilot exercise is being set up using Year 2 software.

Year 3: The experience obtained using Year 2 implementation will be taken into account in a new (and final) version of the software. The DS protocol will follow as close as possible the emerging ISO/CCITT standards.

2.4. THORN and the ESPRIT IES

Obviously, the THORN objective is to strictly adhere to the OSI model and protocols. All THORN communication layers conform to ISO OSI services up to level 5. In Year 1 and 2, THORN as decided to adopt the transport and session layers provided by the ESPRIT ROSE

project for the industrial partners computer systems, while some of the research partners have used their own session and/or transport. As far as Year 3 is concerned, the project decided that all layers below session included are to be considered background information.

On top of the session layer, THORN has followed the ECMA proposal of using the *Remote Operation Service (ROS)*. ROS has proven to be not only a clean mechanism for both specifying the protocols and for specifying the interface onto OSI services, but also to provide a clean internal interface for the implementation, which makes it easy to port the software onto a wide range of computer systems and to experiment with different implementations of the lower layers protocols. Thus, the project has decided to stabilize an internal ROS interface also for Year 3.

3. THORN IMPLEMENTATION

Since the project has already completed the Year 2 software delivery, we will only briefly mention the Year 1 delivery, to the extent that it constitutes the basis for the Year 2.

3.1. The Year 1 Delivery

Year 1 THORN is basically a (partial) implementation of the ECMA proposal [2]. The DS provides to the application program (called *the user* in the following) a *White Page* and a *Management Service*. The White Page Service associates *names* with *property lists*, that is, set of <property name - property value> pairs. A name is simply an ordered sequence of <name part type - name part value> pairs. Thus, the name space is a slight generalization of hierarchical name space such as the ones offered by conventional file systems. The user can provide the DS with a name specification and a set of property types, and require the White Page Service to return the values of the requested properties. Wildcarding is allowed on the last component of a name, in order to increase the flexibility of the queries.

The Management Service consists of a number of service elements which allow the user to create and modify the name space and the property lists associated to the leaf directory entries. This service includes facilities for adding, modifying and deleting name components, leaf nodes and properties. These service elements are available also from a remote DUA, in order to ease the non trivial name space and property information management problem.

As previously mentioned, Year 1 THORN implemented the DS protocol on top of a ROS. The ROS implementation followed the ECMA specification [5], using a BCS half-duplex session implementation provided by ROSE. The implementation has been improved to negotiate both full- and half-duplex during Year 2. All X.409 encoding/decoding functions have been developed without making use of an X.409 compiler.

Very rough design choices have been adopted for supporting the DS name space, implementing it on top of the Unix(*) available library archiving system.

The major limitations of the Year 1 delivery were the absence of distribution and replication of the *Directory Information Base (DIB)*, the absence of an aliasing service, and of access control. But the limited scale pilot exercise run by the research partners also emphasized that a functionality of "reading the children" of a non-leaf node was absolutely necessary and a more efficient data base tool was required, because the provided one was not able to cope with bulk loading requirements.

* Unix is a trademark of AT&T Bell Laboratories.

It is quite interesting to note that the pilot exercise pointed out functional specifications limitations which were not apparent at the beginning. The need for a name space browsing facility, explicitly supported by the DS, turned out to be a basic one when human users come into the scene. Bulk loading is also a crucial aspect, which must either be addressed by special purpose tools which must carefully be planned, or may be circumvented by the "normal" DS operations, provided that the latter have not been implemented taking too seriously the assumption that update operations are very rare in a DS.

The major goal for the Year 2 of THORN was thus that of overcoming or at least alleviating the above limitations, and to align to the emerging ISO/CCITT standards.

4.1. The Year 2 THORN

The main evolution of Year 2 over Year 1 is the possibility of distributing the DS into several sites, in order to cope with both functional and organizational problems. Thus, a number of DSAs are available to be contacted by a user via its DUA. The distribution of the DIB will help providing a better response time. Since most of the accesses will be to *local information*, exhibiting a clustered access pattern, the DIB needs to be only partially replicated, thus increasing the overall availability of the service and decreasing the cost of the DS. The (partial) replication of the DIB does not introduce all the issues (and poor performances) which are typical of distributed data bases because in DSs instantaneous global commit of updates are not expected.

When the project plans were prepared in 1984, there was a general expectation that a quite stable international standard was to be available by the end of 1985. Instead, it turned out during 1985 that the standardization process became much more promising, because of the involvement of both ISO and CCITT in joint meetings, but also of a much wider scope with quite advanced functional requirements. Thus, at the end of Year 1, THORN had no possibility of targeting at an emerging standard, which had to be relatively stable and of complexity to be tackled within the budget of THORN. We decided to use the ECMA based Year 1 design and to extend it.

The general principles behind the Year 2 THORN design and implementation are the following:

1. Allow several DSAs with incomplete DIB.
2. Replicate the DIB at the level of complete sub-trees, and distinguish *the master copy* of a sub-tree from the *shadow copies*.
3. Do not require instantaneous commit of updates, but guarantee that updates are made on master copy only, and in a finite time *propagated to shadows*.
4. Use *hinting* only for transferring an operation to the correct DSA: no *chaining* nor *broadcasting* among DSAs.
5. Allow DUAs to choose one out of several DSAs for first contact, and one out of several DSAs for hint following.
6. Completely separate user programs from DUA service, i.e. use linked libraries to access DUA from the user program.
7. Use Unix Version 7 facilities only, to provide wide portability of the developed code.

In addition to the main extension for replication, distribution and navigation, *new services* have been made available related to *aliasing*, to *management of group properties*, i.e. multiple values for properties, and to *access control*, and finally the *response time* of the service has been optimized by using a direct access method to secondary storage.

In the following section, we will describe some relevant aspects of the achievements of Year 2.

3.2. Client and Server Architecture

As far as the client side is concerned, the two classical alternatives are to implement the DUA either as a process, which serves all the requests coming from the user processes existing on the system, or as a run time support library (hopefully shared among all the user processes), which runs in the context of the calling user program.

While in Year 1 the first alternative was adopted in order to multiplex session connections, to save program space and to batch the requests to the DSA, in Year 2 we moved to the second solution in order to optimize the response time to the single user request, and to be closer to the CCITT emerging proposal.

At the server side, a DSA process is spawned (by the session service) for each incoming ROS call: the level of parallelism can be controlled and the readers/writer problem is correctly handled. Each DSA process handles one connection at the time, but may perform more than one operation per instantiation. The DSA process terminates when the DUA releases the ROS association.

The user interface procedures map one-to-one to the defined *Directory Access Protocol (DAP)* elements. Thus, the DUA performs the X.409 encoding, makes use the ROS interface to forward the call, and decodes the results received via ROS. *Ad hoc* X.409 coding/decoding functions have been implemented, which encode the DS types as indefinite length X.409 constructors.

DUA also handles DSA unavailability by trying a configurable list of *first contact* alternative DSAs, before reporting failure. The DUA is also in charge of navigation, as we will describe in the appropriate section, since we have adopted the hinting method. Finally, DUA logs type, completion and time of all operations on a per system file, which can be the basis for statistics collection and for service charging.

The DSA also is obviously based on the DAP elements, and roughly speaking performs one DIB access per operation requests. The sources of additional DIB accesses are aliases, user identifications and access right checks.

3.3. Data Base Implementation

The DIB is implemented on top of an extension of the Unix public domain *dbm* package. For each DIB there is *name space dbm* and an *object dbm*. The object dbm stores all the leaf entries of the DIB, which, according to the ECMA proposal, are the only nodes which may contain information. The name space dbm instead stores the intermediate node information that is needed by the DS to perform the *ReadChildren* operation, and to lock parts of the DIB on updates.

The dbm package maps records consisting of a variable length key plus a variable length data part onto a sparse file, exploiting the block allocation-on-demand policy of the Unix file

system(*). The *distinguished name* of the entry is used as key to a data record describing the entry content, both for name space and object dbms. Extensions to the basic dbm are needed, because otherwise the (key+data) record must fit into a single dbm page, clashing records must fit into a single dbm page, update is not supported, and finally dbm does not inform the caller whether the dbm page is full on unsuccessful queries. Extensions are based on subdividing long records into *packets*, whose key is an extension of the basic record key. The clashing problem has been solved by providing re-hashing (again based on key extension) both on insertion and (unfortunately cannot be avoided) on query. The extended dbm functionality provides to the upper layer a very simple associative memory, which can easily be re-hosted on a different access method.

3.4. Distribution, Replication and Navigation

The global Directory Service Information Base can be partitioned and distributed at sub-tree level. As a limitation for Year 2, level one sub-trees only can be distributed. Each sub-tree is mastered by a unique DSA, but can be slave-copied by one (several) DSA(s). There is no limit to the number of sub-trees that a DSA can master and/or slave-copy. The set of sub-trees managed by a DSA constitutes the DSA DIB.

A unique sub-tree, named *the Table*, describes the DS partitioning, namely which DSAs are there, which sub-trees they master and which ones instead they slave-copy. To try simple things first, we decided that the *Table* is mastered by no DSA and is slave-copied by all DSAs.

Read operations can be performed on any copy of a sub-tree, possibly the most convenient to the user. But a contacted DSA can refuse to perform the requested operation, and suggest the DUA to contact another DSA if and only if:

1. The operation would require the modification of a sub-tree the DSA does not master: In such a case the DSA suggests the DUA to contact the master of the involved sub-tree.
2. The operation requires the availability of information the DSA does not have: In such a case the DSA suggests the DUA to contact the slaves or the master of the involved sub-tree.

Thus, a DUA may receive from the DSA a refusal to perform the operation with a *hint to contact* other DSAs for the operation. The hinting information allows the DUA to access again the DSA to obtain the names of the DSAs that may help. Before requesting the addresses of them, the THORN DUA intersects such a list with a configurable list of allowed DSAs. This feature allows to configure a control on the amount of resources that can be spent in trying to perform an operation requested by the user and to specify an order of preference in following the hint. In fact, it is worth noting here that both the process of trying alternative first contact DSAs and the process of navigating the DS by hint following are performed by the DUA code without any intervention of the user program.

As a concluding remark, let us emphasize that the master DSAs only may perform modify operations. Slave copies must be updated by the master DSA, as we will see in a following section.

* This is the only design choice which introduces a Unix dependency.

3.5. Access Control

Access control has been the second major extension of Year2 over the ECMA based Year 1 implementation. The level and sophistication of the introduced access control is quite limited, but it is nevertheless a first step towards an adequate level of service.

The THORN access control scheme is centered around the idea that read access is granted to everybody, also to non registered users. Modify operations instead can be protected, and granted only to a specifiable list of users, to be identified by means of a weak identification scheme, i.e. via a password. This introduces into the DIB *sensitive* properties, namely *password*, to store the user password, and *Access Control List (ACL)*, whose value is the list of the names of the users who are allowed to perform any modify operation on the entry which contains such an ACL property (an empty or non existing ACL means that no modify operation is allowed on the DIB entry). These properties are sensitive because the read access to them must also be protected. The scope of ACL is extended to protect the read access of sensitive properties of the entry.

From the above description it is clear that a new entry must be appropriately initialized with a non empty ACL. Moreover, we do not want that casual users can modify the name space by introducing new entries, or even intermediate nodes in the DIB. It is thus convenient to introduce the notion of *DS managers*, (registered in the *Table*) who are allowed all operations. As we will see later, it is also useful to trust all operations requested by remote DSA processes. This special kind of user is identified via the session address of the incoming call. In other words, DSAs are considered as DS managers and are trusted in all situations. This is certainly not a completely secure protection, because it relies on the security of a foreign computer system and on the security of the session connection.

3.6. Consistency Maintenance

A THORN DSA is responsible for the propagation of all the updates performed onto the DIBs it masters. A master DSA keeps a log file for each mastered DIB, and periodically tries to contact all the DSAs which have a slave copy of such a DIB, and to propagate the collected updates. Since slave DSAs may be unavailable, and the batch propagation may abort before completion, a file is maintained for each slave DSA in which *still pending updates* are accumulated in the correct order.

The propagation of updates does not require the introduction of *Directory Service Protocol (DSP)* elements. In fact, the master DSA just invokes on the slave DSA the execution of exactly the same modify operation that it executed successfully, and that the slave DSA would normally refuse to perform because it modifies a slave copy. The only extension we need consists in identifying DSAs as DS managers.

3.7. DS Management Utilities

Year 2 THORN provides a first step towards an adequate DS management support. All the update propagate functions are supported via *ad hoc* utilities (shell commands) for "distributing" the global log file into the DSA specific pending updates and for propagating the pending update file to the involved DSA. Thus, DS managers can freely choose (typically via *cron*) the frequency and time for running such a service, on a per DSA basis.

DS managers must also be supported in saving the state of a DIB and restoring it. Such a restore function should also be available as a remote service, so that a slave DSA can restore a "crashed" slave copy, and re-synchronize it to the evolving master copy. In order to

provide such functionality, Year 2 THORN introduces two DSP elements called *RequestSubTree* and *TerminateRequest*, both to be issued by the slave DSA to the master one.

RequestSubTree returns to the calling DSA the name of a file in which the master DSA has stored the dump of the requested sub-tree. The slave DSA can then fetch this file with any available tool, typically a file transfer service. Once that the slave DSA has obtained the sub-tree dump, restored it in its local DIB, it must issue a *TerminateRequest* call, to inform the master that the new copy has been put on line. In this way, the master DSA can stop propagating changes to a slave which is not working correctly, accumulate them, and perform the propagation when the situation at the slave site is stable, thus avoiding loss of update propagations. The dependencies between update propagation and sub-tree dump-and-restore function are taken into due account, to avoid losses of information in slave copies, and the restore function can also support initial installation of a slave copy.

Finally, DS managers must be supported in the process of setting up the name space of a sub-tree, and in managing the *Table*. Year 2 THORN provides a *DSadmin* utility which offers to the DS manager all DS operations on its local DIB.

3.8. Preparing the Large Scale Pilot Exercise

In Year 1 the research partners have harmonized their X.400 electronic mail systems, which in this way interconnect various national networks. This preparatory work lays down the basis both for the cooperation among the THORN partners, which heavily (and solely) depend on electronic messaging for exchanging reports, documents and comments, but also for the large scale experiment to run on the Year 2 software. The objectives of such an experiment are:

1. To provide significant utilization and testing of Year 2 software. The level of use should be as high as practicable.
2. To test, and gain experience with, update, query and management facilities.
3. To gain experience with replication and distribution of DS data.
4. To gain experience with a directory service containing *real* data, and accessed by *real* users (as opposed to simply be a demonstrator).
5. To provide a feedback into the Year 3 specification, design and implementation of THORN.

A possible way to use a directory service is via a simple screen oriented user interface, so that the user can directly query the directory. Such a development as been completed during Year 2, and it has proven very useful, also for demonstration purposes. More interesting, work is in progress to integrate the use of the DS into applications, so that for example:

- A mail router can find the address of a remote MTA by an automatic DS search.
- A distribution list expander can rely on the DS for distribution list updating and expanding.
- The end user can designate its correspondats by user friendly DS names, possibly using DS aliases; the electronic mail UA will access the DS before formatting the message, without user intervention.

The research partners that are in charge of setting up the experiment are preparing bulk loading tools to set up a DS containing real data about real users. The DS will contain data of the following nature:

1. Human information. Telephone Numbers, O/R Names, and similar information will be associated with human users.
2. Organisational information. Telephone numbers, and other information, associated with organisations, and parts of organisations.
3. Distribution lists. Lists of users. The lists will be of names contained within the directory service, so the list may be used for a number of purposes, although the primary use is expected to be X.400. This mechanism may also be used for handling Roles. Every Organisation and (optionally) Organisational Unit(s) will have an associated list 'postmaster'.
4. MTA information. Information relating to MTAs in the context of an organisation will be provided. This may be for direct access by the MTS, or for access by a system manager to update local routing tables.
5. X.29 information. Information on X.29 services will be provided. This is intended for use by humans, rather than for X.29 software.

A quite large community of users is expected to use THORN, since the involved organizations are the ones that participate into the project. The research partners are planning to offer the THORN service on an experimental basis to most of their members, and to users connecting to their services.

4. THORN EVOLUTION

The project is now in the middle of the Year 3, which according to initial schedule is supposed to be the final one. Even if it has accumulated some limited delays, due to internal problems, the project has to face an instability of the emerging standards. As mentioned above, the joint work of ISO and CCITT has provided quite an increase of scope in the standardization process, which went through a "high" of a very complex proposal, to the "down" of a "strawman" one, and now is hopefully finalizing the definition process which is expected to be substantially over by the end of 1987.

In the beginning of Year 3, THORN is proceeding to complete the specification of the final delivery depending on its previous experience, the results of the initial phases of the Large Scale Pilot Exercise and, more important, on the output of the joint meetings of Egham and Tokyo. A first delivery of Year 3 software based on such a specification is expected by the first quarter of 1988, and a final one for few months later, which can accommodate the final minor modifications which are expected on the emerging standards. If the promise of stability of the current standard proposal are kept, THORN can provide an early, almost compliant, implementation of the Directory Service Standards.

It is worth noting that, nevertheless, the implementation of the ISO/CCITT emerging standard appears to be far more complex than the original ECMA proposal, so that delays on the anticipated three year time span, and additional development costs must be expected.

6. CONCLUSIONS

The THORN project has proven that it is viable to develop a Directory Service adequate to the ESPRIT community needs. Improvements are certainly still needed on functionalities, performances, and robustness, but the basic capabilities are already present in Year 2 software.

Working with an evolving situation in the field of international standards has proven to be certainly very hard, but quite stimulating. We think we can say this for sure on the side of the implementors, but hopefully a project like THORN might be useful also for the standardization bodies.

REFERENCES

- [1] Oppen, D.C. and Dalal, Y.D., The Clearinghouse: A Decentralized Agent for Locating Named Objects in a Distributed Environment, Xerox Office Product Division Internal Report OPD-T8103, October 1981, Xerox, Palo Alto.
- [2] Directory Service Standard (final draft) ECMA-xx TR23/84/110 (July 1985).
- [3] CCITT/ISO, Directory System Standard, X.ds|DP9594/1-8, Tokyo Version (June 1987).
- [4] Huitema, C. and Kille, S., THE Obviously Required Name-server (THORN): Progresses and results, THORN Internal Report INRIA-12, Inria, Rocquencourt (July 1986).
- [5] Remote Operations: Concepts, Notations and Connection-Oriented Mappings, ECMA-yy TR32/85/174 (July 1985).

INDEX OF AUTHORS

- ADAMS, A.E., 1707
 ADER, M., 1205
 AGNEW, M.J., 24
 AINGER, A., 1795
 ALTY, J.L., 819
 ANDERL, R., 1570
 ANDERSEN, M.B., 127
 APOSPORIDIS, E., 736
 ARLABOSSE, F., 909, 985
 ARROWSMITH, P., 82
 ARSAC, P., 798

 BACCARINI, G., 299
 BAKER, S., 1413
 BARFOOT, K.M., 55
 BARNARD, P., 1101
 BATHE, C., 1459
 BAYERL, S., 721
 BELLIA, M., 771
 BERKELAAR, M., 195
 BESLMUELLER, E., 1077
 BEVAN, D.I., 701
 BIERMANN, J., 909
 BIGHAM, J., 936
 BLAND, S.W., 24
 BLANG, G., 148
 BLUMANN, W., 1822
 BLYTHE, J., 630
 BOES, U., 1325
 BOILLON, D., 1175
 BOIS, D., 33
 BOLAND, F., 271
 BOLSENS, I., 251
 BON, M., 103
 BOOTH, J., 1533
 BORASIO, C., 1765
 BORCHERS, H.W., 973
 BORRON, H., 1276
 BOSCO, M., 432
 BOSCO, P.G., 771
 BOSMAN, D., 1288
 BRADIER, A., 750
 BREUKER, J., 963
 BRYER, N.J., 127
 BUDKOWSKI, S., 543
 BUETHER, H., 1175
 BURN, G.L., 701
 BUTLER, J.W., 936
 BYERLEY, P., 1101

 CAHILL, C.G., 55
 CAMPBELL-GRANT, I.R., 1378
 CARR, D., 103, 1101
 CASALE, G., 375
 CASERTA, S., 1077
 CAULFIELD, B., 1413
 CAVANNA, A., 985
 CELENTANO, A., 1005

 CHANDLER, N., 82
 CHAPUIS, D., 55
 CHEDGEY, C., 1583
 CHOUAN, Y., 148
 CHRISTODOULAKIS, D., 361
 CHRISTOU, A., 159
 CLAESEN, L., 251
 CLARK, M., 1662
 CLARK, M.G., 127
 CLARK, W.J., 1149
 CLOAREC, J.-F., 339
 CLOETENS, L., 241
 COCITO, M., 271
 COCKRILL, J., 29
 COCKY, J., 251
 COLASSE, J., 1822
 COLMERAUER, A., 611
 CONRATH, D.W., 1077
 CONTI, P., 1389
 COOK, T., 1822
 CORD, B., 164
 COSIMA PROJECT TEAM, 1648
 COURTIN, P., 985
 COURTOIS, B., 271
 COXON, P.A., 127
 CROCHET, M.J., 113
 CULLEN, J., 1053
 CULLEN, K., 1053
 CUNNINGHAM, R.J., 566

 DAELEMANS, W., 1226
 DALE, C., 415
 DAVID, J., 1161
 DAVIES, D.G., 1443
 DAVIES, I., 29
 DE ANTONELLIS, V., 1066
 DE MAINDREVILLE, C., 640
 DE MAN, H., 207, 251
 DECONINCK, P., 113
 DEGLI INNOCENTI, M., 375
 DEHAINE, G., 82
 DEHM, B., 503
 DELKER, K., 11
 DEPEY, M., 19
 DESOYER, J.C., 1188
 DIAZ, M., 543
 DIEFENBACH, J., 148
 DOLPHIN, C., 1053
 DOREE, P.R., 1693
 DOSTER, W., 1325
 DUMAS, P., 1077
 DUPRET, F., 113
 DUSEAUX, M., 113
 DWOLATZKY, B., 1638

 ELZER, P., 973
 ERL, M.J., 521
 ERNST, G., 1608
 EYRES, P.K., 1745

- FAEHNRIK, K.-P., 1443
FAHRENSCHON, K., 127
FARHOODI, F., 1662
FARRELL, S., 1517
FASANO, F., 1378
FERRARI, G., 375
FIGUEIRAS, J., 426
FISCHER, H.C., 535
FOGARASSY, E., 1188
FOGAROLI, G., 1325
FONIO, H.-R., 503
FOSTER, A., 1101
FOUGERE, M., 1517
FOWLER, T., 1101
FUCHS, C., 1188
- GARCIA, J., 1265
GARIBOTTO, G., 850
GAUSSENS, E., 909
GENGENBACH, U., 1547
GEORGE, C., 451
GERLACH, H., 503
GHANI, N., 1707
GIACHIN, E., 836
GIAVITTO, J.-L., 480
GIBELLI, M., 432
GIOVANNETTI, E., 771
GIRARD, B., 1533
GLASL, A., 11
GODBERSEN, H.P., 1459
GONZALEZ-RUBIO, R., 750
GOODFELLOW, R., 103
GOUBERT, J., 241
GOUTAS, S., 361
GRABOWSKI, H., 1570
GREEB, K.H., 127
GROPPETTI, R., 1765
GUEBELS, P.P., 241
- HAAG, J.P., 1161
HAGE, P.M., 1707
HAGGENMUELLER, R., 535
HARP, J.G., 583
HART, P.A.H., 3
HATZOPOULOS, Z., 159
HAYSelden, B., 325
HAYWARD, S.A., 665
HEFNER, A., 19
HEINZ-FISCHER, H., 1619
HILL, P., 1124
HIRSCHHEIM, R., 1041
HODGSON, G.S., 521
HOEKSTRA, R., 1413
HOFER, R., 218
HOLLNAGEL, E., 811
HOLVOET, Y., 480
HORN, C., 1490
HUDSON, A.P., 1137
HUISKEN, J., 207
HUNT, P., 19
HUSSLA, I., 72
- ISENBERG, R., 1671
- JARKE, M., 402
JENSEN, F.R., 891
JESS, J.A.G., 195
JONCKERS, V., 1226
JOUBERT, P., 148
JOURDAN, M., 1265
- KARAPIPERIS, L., 55
KARIA, R.J., 701
KELLY, W.M., 103
KERBER, W., 1091
KIEBACK, A., 1091
KIERNAN, G., 640
KILLE, S.E., 1472
KILLICK, R.D., 1816
KING, S.G., 936
KLEIN, E., 867
KLEINE, U., 218
KNAUER, K., 218
KNIFFLER, N., 1188
KOETHER, E., 1378
KOK, J., 521
KORIBA, M., 1745
KOULOUMDJIAN, J., 593
KOYMANS, R., 311
KRAKOWIAK, S., 1490
KRAMMER, M., 148
KRIEG-BRUECKNER, B., 491
KROENERT, G., 1367
KUIPER, R., 311
KUNTZMANN, A., 426
KURZWEIL, K., 82
- LARSEN, H.L., 688
LE PESANT, J.P., 127
LEFEVRE, J.P., 1175
LEITCH, R., 985
LETZ, R., 721
LEVI, G., 593, 771
LEVIN, M., 985
LOEBL, H., 1325
LOEHNERT, K., 113
LOISEL, B., 148
LOPEZ, M., 1413
LORIMY, B., 1777
LOS, R., 1619
LUVISON, A., 1477
- MAGARINO, J., 127
MAKH, S.S., 1501
MALUENDA, J., 113
MARITSAS, D., 361
MARTIN, G.H., 113
MARTIN, P., 790, 985
MASLIN, G., 1325
MAUBOUSSIN, A., 480
MCALLISTER, B., 29
MCALPINE, G., 1425
MEHRING, P., 736

- MEILING, E., 466
 MEITNER, H., 1501
 MELGARA, M., 271
 MERMET, J.L., 55
 MEYER, W., 1671
 MIDDELHOEK, M.G., 559
 MIGLIORATO, P., 127
 MODESTI, M., 593
 MOELLENBACH, K., 1111
 MOISO, C., 771
 MOREL, O.R., 1378
 MORIN, F., 127
 MURCHIO, P., 1765
 MURPHY, S., 1795
- NAFFAH, N., 1251
 NAGEL, G., 113
 NEBBIA, L., 1175
 NEEDHAM, D., 630
 NESS, A.J., 1501
 NICODEME, P., 113
 NIJS, J., 1188
 NONNENGART, A., 566
- PACINI, G., 375
 PAETZOLD, B., 1570
 PALAMIDESSI, C., 771
 PAOLINI, P., 1005
 PAPANICOLAOU, N., 159
 PATTYN, H., 1188
 PATUREAU, J.P., 1594
 PAUTHE, P., 480
 PEDERSEN, G.S., 688
 PERCY, R.A., 1501
 PETERS, J., 874
 PETERSEN, A.B., 1111
 PIETRI, F., 375
 PLACENCIA PORRERO, I., 1300
 POELS, H., 1619
 POHL, C., 1619
 POLS, L.C.W., 1175
 POORTMANS, J., 1188
 PRESTON, N., 392
 PROCTOR, G., 271
 PULESTON JONES, J., 24
 PUNCELLO, P.P., 375
- QIAN, Z.-M., 1188
- RABAAY, J., 207
 RABITTI, F., 1389
 RATHGEB, M.R., 1413
 REGOLINI, J.L., 55
 REIM, F., 1501
 REMY, C., 1822
 REYNAERT, PH., 251
 REZAZADEH, A., 103
 RINGEL, J., 1619
 RIZK, A., 1265
 ROBINSON, P.J., 1378
 ROCHE, M., 19
 ROHMER, J., 750
- RONAYNE, T., 1053
 ROULLET, G., 1477
 RUDE, S., 1570
 RUGGIU, G., 1517
 RULLENT, C., 836
 RUSSELL, P., 1777
 RYAN, G., 1053
 RZEP CZYNSKI, Z.G., 1707
- SAFFIN, R., 1101
 SANDBERG, J., 963
 SANOFF, S.P., 1638
 SCHAEFER, G., 1041
 SCHEER, H.-C., 72
 SCHLECHTENDAHL, E.G., 1547
 SCHMOLLA, W., 148
 SCHOEN, K.R., 11
 SCHOULER, P., 19
 SCHROOTEN, G., 251
 SCHUMACHER, G., 535
 SENSKE, W., 127, 148
 SEVERYNS, R., 251
 SHACKEL, B., 1019
 SHEPPARD, M., 1413
 SIEBERT, H., 973
 SIMON, E., 640
 SIMONE, C., 1066
 SIROVICH, F., 1838
 SIX, P., 251
 SMAILES, R., 72
 SMITH, D.A., 55
 SOLA, D., 1765
 SOMMER, D.H., 1807
 SOMMER, W., 503
 SOREL, V., 1517
 SOUPOS, P., 361
 SPUR, G., 1716
 STANDEN, G., 1413
 STAUSHOLM, L., 826
 STEFANINI, A., 985
 STEPHENS, E., 1517
 STIENEN, H., 1619
 STOK, L., 195
 SUNDSTROEM, G., 811
 SZALAS, A., 566
- TEDD, M., 334
 TEXIER, M., 1251
 THEEUWEN, J.F.M., 195
 THOMAS, G., 1822
 THONEMANN, H.G., 279
 TOBIASCH, R., 503
 TOFT, F., 1477
 TORRIGIANI, P., 375
 TRILHE, J., 42
 TRUCKENMUELLER, T.W., 1413
 TUENI, M., 1205
 TURINI, F., 375
- V.D. BORN, R., 195
 VALENT, R., 339
 VAN DAELE, J., 1188

VAN DER BAAREN, J., 955
VAN DER MEULEN, A.E., 1313
VAN MARCKE, K., 1226
VAN MEERBERGEN, J., 207
VAN NES, F.L., 1452
VAN REEWIJK, C., 559
VAN VYVE, J.-M., 1239
VANDEN MEERSCH, E., 251
VANDEWEERD, I., 218
VANNER, K., 29
VENKEN, R., 402
VERNAY, Y.J., 271
VICI, A.D., 891
VISSERS, C., 543
VITTORELLI, V., 1358

WAGNER, G., 287
WARD, G., 1101
WEBSTER, D.E., 72
WEIR, G., 811
WEIR, G.R.S., 819
WEISANG, C., 973
WERNER, W.M., 11
WIEDER, A., 3
WIESINGER, K., 11
WILLIAMS, F., 1517
WILLIAMSON, G.I., 936
WINKELS, R., 963
WINTER, B., 1822
WINTER, D.T., 521
WIRZ, P., 164
WITTIG, T., 909
WRIGHT, P.H., 1341
WYNNE, R., 1053

YOUNG, N., 1124

ZAROLIAGIS, C., 361
ZIEGLER, J., 1443
ZIJLSTRA, E., 311
ZINSER, K., 973
ZYSS, J., 178

PROJECT ACRONYM INDEX

Acronym	Proj. No.	Page
ACCORD	(P1062)	1594
ACORD	(P393)	867
ADKS	(P311)	874
ADVICE	(P271)	271
AIDA	(P888)	279
ALPES	(P973)	392
AMICE	(P688)	1777
APSYS	(P401)	375
BICMOS	(P412)	3
CAD-I	(P322)	1547
	(P322)	1570
CARLOS	(P718)	1807
	(P718)	1816
CNMA	(P955)	1533
COCOS	(P956)	1251
	(P956)	1265
	(P956)	1276
COMANDOS	(P834)	1490
	(P834)	1501
COSIMA	(P477)	1648
CVS	(P802)	287
DAIDA	(P892)	402
DELTA-4	(P818)	790
DESCARTES	(P937)	311
DIAMOND	(P1072)	521
	(P1072)	535
DMA	(P940)	850
DOEDIS	(P231)	1413
EPSILON	(P530)	593
EUROHELP	(P280)	826
	(P280)	955
	(P280)	963
FAOR	(P56)	1041
FORFUN	(P881)	559
FORMAST	(P1033)	566
GRADIENT	(P857)	811
	(P857)	819
	(P857)	973
GRASPIN	(P125)	361
	(P125)	503
HERODE	(P121)	1367
HUFIT	(P385)	1019
	(P385)	1443
	(P385)	1452
ICD	(P991)	195
IMPW	(P938)	432
INCA	(P395)	1459
	(P395)	1472
INDOC	(P1542)	1005
INSTIL	(P1063)	630
ISIDE	(P1133)	640
IWS	(P82)	1205
	(P82)	1226
KIWI	(P1117)	688
KNOSOS	(P974)	329
LION	(P169)	1477

Acronym	Proj.No.	Page
MARS	(P998)	1517
METEOR	(P432)	480
MIAC	(P1057)	1149
MINSTREL	(P59)	1425
MULTOS	(P28)	1389
OSSAD	(P285)	1066
	(P285)	1077
PADMAVATI	(P967)	798
PALAVDA	(P415)	701
	(P415)	721
	(P415)	736
	(P415)	750
	(P415)	771
PODA	(P1024)	1378
PROMINAND	(P878)	1111
PROSPECTRA	(P390)	491
RAISE	(P315)	451
	(P315)	466
REQUEST	(P300)	415
ROSE	(P33)	1822
SAPPHIRE	(P1277)	325
	(P1277)	334
SEDOS	(P410)	543
SFINX	(P1262)	352
SIP	(P26)	836
SMART	(P1609)	426
SOMIW	(P367)	1239
SPECTRE	(P554)	33
SPIN	(P64)	1175
TODOS	(P813)	1091
WSI	(P824)	42

KEYWORD INDEX

Keyword	Proj.No.	Page
3D algorithms	(P962)	299
3D modelling for manufacturing purposes	(P409)	1608
3D SOI	(P245)	55
3D stereo	(P940)	850
A4 flat panel display	(P833)	127
abstract data types	(P496)	108
abstract data types (ADT)	(P1133)	640
abstract syntax tree	(P125)	361
ACCORD	(P1062)	1594
accurate arithmetic	(P1072)	521
	(P1072)	535
acquisition by example	(P1063)	630
Action plan generation and monitoring	(P82)	1205
active-matrix	(P833)	127
activity framework	(P56)	1041
actors	(P285)	1077
	(P878)	1111
acyclic graphs	(P311)	874
Ada	(P390)	491
	(P496)	108
	(P937)	311
	(P956)	1265
	(P1033)	566
	(P1277)	325
Ada arithmetic	(P1072)	521
Ada packages	(P496)	108
adaptable browsing tools	(P432)	480
adaptive binary arithmetic coding	(P563)	1137
adaptive discrete cosine transform	(P563)	1137
adaptive interpolators	(P97)	207
advanced information processing	(P384)	1638
ADVICE	(P271)	271
AI in CIM	(P418)	1662
Airy disc	(P612)	1288
Alexander method	(P415)	750
alfa conversions	(P415)	771
algebra of programs	(P390)	491
algebraic semantics	(P390)	491
algorithmic parallelism	(P1085)	583
ALICE	(P967)	798
ALPES	(P973)	392
aluminium	(P554)	33
aluminium etch modeling	(P574)	72
Alvey-ISF	(P1262)	352
amorphous silicon	(P491)	148
	(P1051)	1188
amorphous silicon TFTs	(P833)	127
analog cells	(P802)	287
analog circuit	(P881)	559
anaphora	(P393)	867
ANNA	(P390)	491
ANSA project	(P818)	790
	(P834)	1490
ANSI Y14.26M	(P322)	1547
application generator	(P477)	1648
application network	(P477)	1648
application profile	(P1024)	1378
application profiles	(P121)	1367

Keyword	Proj.No.	Page
application specific ICs	(P97)	207
architectural synthesis	(P97)	241
architecture	(P26)	836
architecture of software development	(P432)	480
arithmetic	(P1072)	521
	(P1072)	535
arithmetic expression evaluation	(P1072)	535
ARITHMOS	(P1072)	535
arm	(P940)	850
ASDL	(P125)	361
ASN.1	(P1024)	1378
assembly	(P384)	1638
assembly equipment	(P384)	1638
assessment	(P878)	1111
assistant	(P857)	819
assistant systems	(P1117)	688
associative machine	(P967)	798
asymetric ciphers	(P998)	1517
asynchronous processes	(P1033)	566
ATMOS	(P962)	299
attribute grammars	(P956)	1265
attributed relational graphs	(P28)	1389
audio conference	(P1057)	1149
audio visual interaction	(P1057)	1149
authentication	(P719)	1838
authoring systems	(P901)	1124
automated assembly	(P812)	1765
automatic/assisted design validation	(P271)	271
AVALON	(P387)	909
	(P387)	936
BACK system	(P311)	874
BALDRIC	(P387)	909
	(P387)	936
baseband	(P955)	1533
BBC Master Series computers	(P901)	1124
BBN Butterfly	(P967)	798
benefits analysis	(P56)	1041
BIM Prolog	(P892)	402
binary rules	(P820)	985
bipolar	(P802)	287
bipolar CMOS	(P412)	3
bipolar technology	(P281)	11
bit-parallel architecture	(P97)	207
bit-serial architecture	(P97)	207
blackboard architecture	(P387)	936
blackboard systems	(P387)	909
	(P932)	1671
block structured language	(P179)	1693
boolean algebra	(P1106)	611
boolean optimization	(P991)	195
Bosch rho2	(P278)	1707
boundary representation	(P322)	1547
brightness in panel display	(P612)	1300
broadband	(P395)	1459
	(P955)	1533
browsers	(P59)	1425
	(P280)	955
	(P432)	480

Keyword	Proj.No.	Page
bumped tape TAB	(P958)	82
bus topology	(P169)	1477
business model	(P688)	1777
C-BAT	(P909)	1745
C-PROLOG	(P415)	771
CACTUS device	(P718)	1816
CAD	(P179)	1693
	(P802)	287
	(P962)	299
	(P973)	392
	(P991)	195
	(P1058)	251
	(P1062)	1594
CAD architecture	(P1058)	251
CAD environment	(P271)	271
CAD functions for CAP	(P409)	1608
CAD interfaces	(P322)	1547
CAD systems	(P322)	1570
	(P623)	1716
CAD workstations	(P1062)	1594
CAD*I	(P322)	1570
CAD/CAM integration	(P278)	1707
CAD/CAP data exchange formats	(P409)	1608
CAD/CAP integration	(P409)	1608
CAMELEON symbolic environment	(P97)	241
camera calibration	(P940)	850
CARLOS project	(P718)	1816
carrierband	(P955)	1533
CASE DIS 8649	(P718)	1807
CATHEDRAL	(P97)	207
	(P97)	241
CATHEDRAL-II	(P1058)	251
CCITT H110-120-130	(P925)	1161
CCITT SGXV OKUBO	(P925)	1161
CCITT T400 standard	(P121)	1367
CCITT Y221 recommendation	(P1057)	1149
CD-ROM	(P901)	1124
CeBIT 87 and 88	(P1024)	1378
cell fabrication	(P833)	127
cell generator	(P802)	287
Cell/Tissue	(P387)	909
cement manufacturing KBS	(P820)	985
CEPT TR1-NA3 Frame structure	(P1057)	1149
certainty	(P59)	1425
character text	(P1024)	1378
Checklands Soft System Methodology	(P56)	1041
chemical industry in IT	(P443)	178
CHILL	(P937)	311
chip bumping	(P958)	82
chip die attachment	(P958)	82
Chomsky	(P311)	874
CIM	(P384)	1638
	(P409)	1608
	(P955)	1533
	(P1030)	1053
CIM and AI	(P418)	1662
CIM architecture	(P688)	1777
CIM assessment	(P909)	1745

Keyword	Proj.No.	Page
CIM enterprise model	(P688)	1777
CIM graphics	(P496)	108
CIM OSA	(P688)	1777
CIM shell	(P932)	1671
CIM systems	(P1199)	1795
CIM systems integration	(P812)	1765
circuit description language	(P881)	559
circuit simulation	(P281)	11
CISC architecture	(P956)	1251
class	(P125)	361
clause compilation	(P415)	721
client/server concept	(P395)	1459
CML	(P892)	402
CMOS	(P97)	218
	(P412)	3
	(P554)	33
	(P802)	287
CMOS-SOI	(P245)	55
CNMA	(P955)	1533
coaching	(P280)	963
COBWEB	(P415)	701
coding algorithms	(P563)	1137
	(P925)	1161
cognitive complexity theory	(P385)	1443
cognitive design aids	(P234)	1101
cognitive models	(P280)	955
	(P280)	963
cognitive processing	(P857)	811
cognitive psychology	(P234)	1101
cognitive simulator	(P234)	1101
cognitive studies	(P878)	1111
cognitive task representation	(P385)	1443
coherence	(P285)	1066
colour displays	(P833)	127
colour mapping	(P612)	1313
COMANDOS	(P231)	1413
	(P834)	1490
combinators	(P415)	701
Commodore AMIGA	(P901)	1124
communication KBMS	(P892)	402
communication protocols	(P169)	1477
	(P1057)	1149
communication standards	(P812)	1765
COMPAC/APS	(P623)	1716
compatibility of operations	(P285)	1066
compiler generation	(P956)	1265
compiler optimisation	(P415)	750
completeness	(P410)	543
complex domains	(P1133)	640
complex work environments	(P1030)	1053
complexity	(P387)	936
	(P410)	543
	(P820)	985
	(P958)	82
component based language	(P820)	985
component vector quantisation	(P563)	1137
components	(P300)	415
compression	(P563)	1137
compression techniques	(P28)	1389

Keyword	Proj.No.	Page
	(P925)	1161
computer integrated manufacturing (CIM)	(P932)	1671
concept base	(P96)	891
concept language	(P82)	1226
Conception to use	(P385)	1019
conceptual graphs	(P26)	836
conceptual model	(P813)	1091
CONCORDIA project	(P818)	790
concurrency	(P1033)	566
concurrent computation	(P1062)	1594
concurrent distributed garbage collection	(P415)	701
concurrent office systems	(P285)	1066
concurrent PROLOG	(P415)	771
	(P956)	1251
concurrent sorter networks	(P97)	218
configuration of distributed systems	(P834)	1501
conformance testing	(P955)	1533
Connection Method	(P415)	721
consistency	(P410)	543
constraint notion	(P367)	1239
constraint satisfaction	(P387)	909
constraint systems	(P1106)	611
constraints	(P938)	432
constructive solid geometry	(P322)	1547
contact imager	(P1051)	1188
contour extraction	(P612)	1288
control and indication	(P1057)	1149
control strategies	(P26)	836
control systems	(P384)	1638
	(P809)	1619
control theories	(P387)	909
convolution	(P612)	1313
coordinate transformations	(P179)	1693
COQUAMO model	(P300)	415
CORDIC	(P179)	1693
Corporation for Open Systems (COS)	(P955)	1533
correctness	(P125)	503
	(P410)	543
	(P415)	721
COSIMA	(P477)	1648
cost	(P1609)	426
COST CODEC 211	(P925)	1161
cost-benefit	(P909)	1745
cost-benefit models	(P1030)	1053
Cranfield benchmark	(P623)	1716
critical path computation	(P97)	207
cryptographic key management	(P998)	1517
crystal engineering workstation	(P443)	178
crystal growth	(P1128)	113
CS multiple access/collision detect.(CSMA/CD)	(P955)	1533
CSMA/CD LANs	(P33)	1822
CSP	(P1033)	566
current controller	(P179)	1693
cursive script	(P295)	1341
Czochralski	(P1128)	113
DAFNE methodology	(P401)	375
DAMES	(P974)	329
data description model	(P432)	480

Keyword	Proj.No.	Page
data driven computer (DDC)	(P415)	750
data encryption	(P998)	1517
data management	(P432)	480
data management system	(P311)	874
data modeling	(P59)	1425
database	(P384)	1638
database management	(P1062)	1594
databases	(P623)	1716
	(P938)	432
	(P1058)	251
DBMS	(P59)	1425
	(P1133)	640
DBMS interfaces	(P418)	1662
DBPL	(P892)	402
decision making	(P1609)	426
decision network	(P932)	1671
decoding	(P925)	1161
deep trench isolation	(P243)	19
Delaunay method	(P962)	299
Delta PROLOG	(P967)	798
DELTA-4	(P818)	790
dependability	(P1609)	426
dependable computer architecture	(P818)	790
dependence rules	(P26)	836
depletion mode transistors	(P843)	29
deposition of a-Si	(P1051)	1188
depth analysis	(P940)	850
description language	(P881)	559
design automation	(P991)	195
design for EDB	(P271)	271
design methodology	(P97)	218
design rules	(P554)	33
	(P802)	287
design to product	(P384)	1638
designer support system	(P385)	1443
despatching	(P477)	1648
deterministic behavior	(P285)	1066
development	(P315)	451
device isolation	(P554)	33
device simulation	(P281)	11
	(P962)	299
DFETs	(P843)	24
diagnosis	(P280)	955
diagnostic modeling	(P280)	955
DIALOG	(P1058)	251
dialogue	(P393)	867
dialogue design	(P857)	819
dialogue interface	(P857)	811
dialogue manager	(P393)	867
dialogue system	(P857)	973
dictionaries	(P64)	1175
	(P280)	826
	(P291)	1358
	(P530)	593
	(P813)	1091
differential PCM	(P563)	1137
digital circuit design	(P415)	736
digital signal processing	(P97)	207
	(P97)	218

Keyword	Proj.No.	Page
digital transmission	(P1057)	1149
diphone templates	(P64)	1175
direct graphic manipulation	(P385)	1443
directory services	(P395)	1472
directory systems	(P719)	1838
disambiguation	(P291)	1358
discourse representation	(P393)	867
discretization	(P962)	299
dislocation	(P1128)	113
dispatching processors	(P967)	798
display architecture	(P612)	1313
display materials technology	(P491)	148
display model	(P612)	1300
display optics modeling	(P612)	1300
display simulator	(P612)	1313
DISSIM project	(P612)	1300
	(P612)	1313
distributed directory systems	(P719)	1838
distributed memory architecture	(P415)	701
distributed office systems	(P834)	1501
distributed systems	(P395)	1459
	(P410)	543
	(P834)	1490
DMOS	(P245)	55
document analogy	(P1542)	1005
document classes	(P121)	1367
document content	(P121)	1367
document editor	(P121)	1367
Document flow analysis (DOFA)	(P285)	1077
document handling	(P295)	1325
document layout	(P367)	1239
document manager	(P395)	1459
document manipulation	(P121)	1367
document preparation	(P395)	1459
document production	(P1542)	1005
document standards	(P121)	1367
document structures	(P121)	1367
document synthesis	(P1542)	1005
document transfer	(P121)	1367
documentation KBMS	(P892)	402
documentation support	(P1277)	334
documents	(P938)	432
DOEOIS project	(P231)	1413
DOFA	(P285)	1077
domain data structure	(P1133)	640
domain layer	(P96)	891
Domesday book	(P901)	1124
double metal process	(P554)	33
drawing	(P973)	392
Dutch linguistic analysis	(P291)	1358
Dutch-NL	(P82)	1205
dynamic concept models	(P813)	1091
dynamic programming	(P991)	195
dynamic scheduling	(P809)	1619
dynamic script recognition	(P295)	1341
dynamic systems	(P857)	819
E-beam debugging	(P271)	271
E-beam evaporation	(P843)	29

Keyword	Proj.No.	Page
E-beam lithography	(P971)	103
EAST project	(P1262)	352
ECL-random access memory	(P281)	11
ECLIPSE	(P1277)	325
	(P1277)	334
ECMA 101	(P395)	1459
ECMA 101 standard	(P121)	1367
ECMA CASE	(P834)	1490
ECMA standards	(P818)	790
ECMA TC32	(P395)	1472
economic factors	(P1030)	1053
economics of CIM	(P909)	1745
edge detection	(P940)	850
EFETs	(P843)	24
electrical debugging	(P1058)	251
electroluminescence	(P612)	1300
electronic mail	(P33)	1822
	(P395)	1459
	(P718)	1807
	(P1024)	1378
electronic mail addressing	(P719)	1838
electronics	(P1058)	251
elimination of redundancy	(P415)	721
ELLA	(P179)	1693
emitter coupled logic	(P971)	103
empirical models	(P820)	985
encryption	(P998)	1517
English linguistic analysis	(P291)	1358
English-NL	(P82)	1205
	(P393)	867
enhancement mode transistors	(P843)	29
Enterprise Network Event 1988 (ENE)	(P955)	1533
entity alphabet	(P28)	1389
environment frame generation	(P432)	480
epipolar transformation	(P940)	850
epistemological description	(P280)	955
epitaxial growth	(P971)	103
epitaxial silicon growth	(P245)	55
epitaxy	(P1270)	159
EPSILON	(P530)	593
equipment model	(P384)	1638
ergonomics	(P385)	1019
	(P385)	1443
ergonomics in CIM	(P1199)	1795
ergonomics of I/O media	(P385)	1452
ESCHER	(P991)	195
ESTELLE	(P410)	543
	(P718)	1816
eta conversions	(P415)	771
etch criteria	(P574)	72
etch homogeneity	(P574)	72
EUCLID	(P409)	1608
	(P623)	1716
EULER	(P991)	195
EUROHELP	(P280)	826
	(P280)	955
	(P280)	963
evaluation	(P64)	1175
	(P1609)	426

Keyword	Proj.No.	Page
evaluation of OIS	(P231)	1413
evaluation taxonomy	(P878)	1111
evaluation transformers	(P415)	701
evaporation	(P334)	164
event graph language	(P820)	985
event model approach	(P857)	819
event parallelism	(P1085)	583
EXAPT language and system	(P409)	1608
excimer laser	(P1270)	159
excitons	(P1270)	159
execution control simulation	(P415)	736
expectations	(P26)	836
experimental boundary file	(P322)	1547
experimental solids proposal	(P322)	1547
EXPERT software system	(P901)	1124
expert systems	(P857)	811
	(P857)	819
	(P857)	973
	(P901)	1124
	(P932)	1671
	(P974)	329
	(P1542)	1005
Expert Systems Builder (ESB)	(P96)	891
expert systems in CIM	(P809)	1619
expert systems shell	(P96)	891
explicit programming	(P623)	1716
Explorer	(P813)	1091
facsimile	(P1057)	1149
factory supervision	(P932)	1671
fail silent controllers	(P818)	790
Fairchild Clipper	(P415)	736
FAOR project	(P813)	1091
	(P878)	1111
	(P1030)	1053
fault detection	(P820)	985
	(P1106)	611
fault diagnosis	(P271)	271
fault dictionary	(P271)	271
fault simulation	(P271)	271
	(P415)	736
fault tolerance	(P818)	790
	(P1609)	426
feature notations	(P82)	1226
FETs	(P843)	24
	(P843)	29
	(P1270)	159
fibre optics	(P169)	1477
field effect mobility	(P491)	148
field trials	(P1030)	1053
Fifth Generation Language	(P956)	1251
file servers	(P395)	1459
file transfer	(P33)	1822
	(P718)	1807
file transfer access and management (FTAM)	(P955)	1533
filing and retrieval	(P59)	1425
finite elements	(P962)	299
first order logic (FOL)	(P415)	721
Flame system	(P96)	891

Keyword	Proj.No.	Page
flat panel display	(P612)	1288
	(P612)	1300
flexible manufacturing system (FMS)	(P809)	1619
floating gate FETs	(P824)	42
floating point operations	(P1072)	521
floor operators	(P1199)	1795
floorplanning	(P802)	287
	(P991)	195
floorplanning methodology	(P97)	241
FM DDL	(P231)	1413
FMS	(P809)	1619
	(P812)	1765
FNC-2	(P956)	1265
focal object	(P231)	1413
FOL	(P415)	721
formal methods	(P315)	451
formal modeling	(P385)	1443
	(P385)	1443
formal office procedures	(P231)	1413
formal techniques	(P410)	543
formatting	(P121)	1367
formatting documents	(P367)	1239
FORTUNE	(P1277)	334
FP2 specification	(P415)	736
fragmentation scattering	(P818)	790
frame formatter	(P367)	1239
frame structure	(P1057)	1149
Freeman encoding vectors	(P295)	1341
French intonation	(P64)	1175
French linguistic analysis	(P291)	1358
French-NL	(P311)	874
	(P393)	867
FTAM	(P718)	1807
functional analysis of requirements	(P1030)	1053
functional circuit language	(P881)	559
functional definition language	(P1133)	640
functional languages	(P415)	701
functional models	(P322)	1570
	(P612)	1288
functional programming	(P415)	750
	(P415)	771
fuzzy rules	(P820)	985
G-MOD	(P387)	936
GaAs	(P843)	24
	(P843)	29
	(P1128)	113
GaAs processing	(P1270)	159
gallium arsenide	(P971)	103
	(P1128)	113
	(P1270)	159
garbage collection	(P415)	701
gate matrix layout	(P991)	195
gateways	(P169)	1477
general object model	(P956)	1251
generalised block truncation coding	(P563)	1137
generalised Petri nets	(P285)	1066
generic office frame of reference (GOFOR)	(P56)	1041
GENESIL	(P179)	1693

Keyword	Proj.No.	Page
GENIE	(P179)	1693
GENTIP	(P962)	299
geometric graphics	(P1024)	1378
geometric modeling	(P1062)	1594
geometric models	(P322)	1570
	(P612)	1288
geometric parallelism	(P1085)	583
German linguistic analysis	(P291)	1358
German-NL	(P311)	874
	(P393)	867
GIE Emerande Sun version	(P1277)	325
Gini algorithm	(P387)	936
GKS	(P295)	1341
glass/TCO/pin/Al structure	(P1051)	1188
glossary of terms	(P385)	1443
GLOT	(P385)	1019
	(P385)	1443
GRADIENT	(P857)	811
	(P857)	819
	(P857)	973
GRAI methodology	(P932)	1671
grammatical categories	(P291)	1358
graphemes	(P291)	1358
graphical browser	(P280)	955
graphical composition	(P1033)	566
graphical dialogue	(P857)	811
graphical expert system	(P857)	973
graphical images	(P28)	1389
graphical user interface	(P432)	480
graphics	(P59)	1425
	(P82)	1205
	(P295)	1325
	(P393)	867
	(P530)	593
	(P612)	1313
	(P813)	1091
	(P857)	811
	(P973)	392
graphics applications	(P1051)	1188
graphics database	(P973)	392
graphics for CIM	(P496)	108
GRASPIN	(P125)	361
	(P401)	375
GRASPIN workstation	(P125)	503
gray levels	(P612)	1313
Greek linguistic analysis	(P291)	1358
gripper management processor	(P278)	1707
grippers	(P278)	1707
guest-host polymers	(P443)	178
handwriting recogniser	(P295)	1325
handwriting recognition	(P295)	1341
Hannover fair	(P955)	1533
health care environments	(P1030)	1053
HEARSAY II	(P26)	836
heat exchange	(P1128)	113
help messages	(P385)	1452
help systems	(P280)	963
hermeneutic stance	(P1030)	1053

Keyword	Proj.No.	Page
heterojunction bipolar transistor (HBT)	(P971)	103
high density TAB	(P958)	82
high level image analysis	(P28)	1389
high resolution display	(P395)	1459
high speed memory cell	(P281)	11
highly secure office systems	(P998)	1517
holistic view	(P878)	1111
homo CVD process	(P1051)	1188
homographs	(P291)	1358
homophenes	(P291)	1358
HOPE	(P956)	1251
Horn clause logic	(P415)	771
HP9000	(P1277)	325
HUFIT	(P385)	1019
human actors	(P878)	1111
human computer interaction	(P385)	1443
	(P385)	1452
human computer interaction (HCI)	(P385)	1019
human factors	(P878)	1111
	(P1030)	1053
	(P1057)	1149
human factors for IT	(P385)	1019
human factors history	(P385)	1019
human factors in CIM	(P1199)	1795
human factors in interfaces	(P385)	1452
human factors in IT	(P385)	1443
IBM PC/AT	(P1277)	325
IC design aids	(P888)	279
icon library	(P96)	891
ID3	(P387)	936
	(P1063)	630
IDEAL	(P415)	771
IES	(P719)	1838
IGES	(P322)	1547
image classification	(P28)	1389
image compression	(P28)	1389
	(P563)	1137
image editor	(P395)	1459
image generation	(P612)	1313
image handling	(P901)	1124
image interpretation	(P28)	1389
image processing	(P97)	218
	(P563)	1137
	(P824)	42
image query language	(P28)	1389
image recognition	(P967)	798
image retrieval	(P28)	1389
image storage	(P28)	1389
image understanding	(P28)	1389
imageur documentaire	(P901)	1124
IMPISH	(P938)	432
implementation	(P315)	451
implicit programming	(P623)	1716
imprecision	(P59)	1425
indirection cell	(P415)	701
industrial control	(P387)	909
	(P857)	811
industrial local area network (ILAN)	(P955)	1533

Keyword	Proj.No.	Page
industrial sub-micron CMOS	(P554)	33
Information Exchange System (IES)	(P33)	1822
information retrieval	(P59)	1425
	(P1117)	688
information system factory	(P1262)	352
information systems	(P59)	1425
	(P938)	432
INGRES	(P1133)	640
inheritance	(P530)	593
	(P956)	1276
initial graphics exchange specification(IGES)	(P322)	1547
ink jet print head	(P295)	1325
INSTIL	(P1063)	630
integrated CAD-EBD system	(P271)	271
integrated circuits	(P971)	103
integrated PAD	(P718)	1807
integrated systems	(P938)	432
	(P1199)	1795
intelligent backtracking	(P967)	798
intelligent help systems	(P280)	826
	(P280)	955
intelligent information systems	(P901)	1124
intelligent monitoring system	(P857)	973
intelligent scheduler	(P418)	1662
intelligent workstation	(P82)	1226
inter MTA communications	(P718)	1816
interaction	(P385)	1443
interactive data intensive applications	(P892)	402
interactive panel display	(P878)	1111
interactive video	(P901)	1124
interconnect	(P243)	19
	(P554)	33
	(P833)	127
	(P958)	82
interface construction	(P956)	1276
interface design	(P385)	1019
international standards	(P563)	1137
interworking	(P169)	1477
investment strategy	(P909)	1745
ISDN	(P563)	1137
ISDN 64Kbit/s	(P925)	1161
ISO 7498 Naming and Addressing	(P395)	1472
ISO 8613	(P1024)	1378
ISO 8879 SGML standard	(P395)	1459
ISO CASE	(P818)	790
ISO Development Environment	(P395)	1472
ISO DIS 8072/8073	(P33)	1822
ISO DIS 8571	(P33)	1822
	(P718)	1807
ISO DIS 8613	(P295)	1325
ISO DIS 8613 standard	(P121)	1367
ISO DP8348/DAD2	(P395)	1472
ISO ODA	(P834)	1490
ISO ODP	(P834)	1490
ISO OSI	(P33)	1822
	(P410)	543
	(P688)	1777
	(P718)	1807
ISO SC21 WG4 MIS standards	(P33)	1822

Keyword	Proj.No.	Page
ISO/CCITT Photographic Experts Group	(P563)	1137
isolated connection	(P415)	721
IT development process	(P385)	1443
IT uptake	(P1030)	1053
Italian intonation	(P64)	1175
Italian linguistic analysis	(P291)	1358
Italian-NL	(P311)	874
IWS project	(P82)	1205
	(P82)	1226
JIT techniques	(P1199)	1795
K-LEAF	(P415)	771
KADS methodology	(P1098)	665
KANBAN just in time	(P932)	1671
KBMS	(P530)	593
	(P892)	402
KBS	(P82)	1205
	(P82)	1226
	(P96)	891
	(P311)	874
	(P387)	909
	(P387)	936
	(P401)	375
	(P820)	985
	(P857)	811
	(P967)	798
	(P974)	329
	(P1058)	251
	(P1098)	665
	(P1117)	688
	(P1542)	1005
KBS assistant	(P82)	1226
	(P857)	819
KBS case studies	(P1098)	665
KBS life cycle	(P1098)	665
KBS scheduling	(P418)	1662
KBS tools	(P1098)	665
KDB	(P938)	432
	(P938)	432
KEE	(P384)	1638
	(P813)	1091
	(P820)	985
	(P857)	819
Kermit	(P1024)	1378
key encryption	(P998)	1517
key management	(P998)	1517
KIRA	(P1117)	688
KIWI	(P1117)	688
KL-ONE	(P311)	874
knowledge acquisition	(P97)	207
	(P387)	936
	(P1063)	630
	(P1098)	665
knowledge base	(P802)	287
knowledge based assistants	(P401)	375
knowledge based simulation	(P932)	1671
knowledge based systems	(P384)	1638
	(P1058)	251

Keyword	Proj.No.	Page
knowledge bases	(P623)	1716
knowledge elicitation	(P1063)	630
knowledge engineering	(P932)	1671
knowledge management system	(P311)	874
Knowledge representation	(P82)	1205
knowledge representation	(P82)	1226
	(P96)	891
	(P280)	955
	(P311)	874
	(P387)	936
	(P932)	1671
	(P1062)	1594
Knowledge Representation System (KRS)	(P82)	1205
Knuth's dimensional concepts	(P367)	1239
Knuth-Bendix algorithm	(P125)	503
Kounalis test	(P125)	503
KRITIC	(P387)	909
	(P387)	936
KRS	(P82)	1226
Kulisch arithmetic	(P1072)	535
lamda-lifting	(P415)	771
LAN	(P169)	1477
Langmuir-Blodgett	(P443)	178
language	(P393)	867
language design	(P956)	1276
language semantics	(P311)	874
large area panels	(P833)	127
large picture bases	(P901)	1124
laservision system	(P901)	1124
layout	(P623)	1716
	(P802)	287
layout process	(P121)	1367
LCD	(P612)	1300
	(P833)	127
LCD displays	(P491)	148
LCD simulation	(P612)	1300
LE-Lisp	(P82)	1226
	(P967)	798
LE-TOOL	(P956)	1251
	(P956)	1276
leaf cells	(P97)	241
learning	(P385)	1443
learning by example	(P1063)	630
lexical ambiguity	(P311)	874
library	(P802)	287
life cycle costing	(P1062)	1594
LIFESPAN	(P974)	329
LiNbO3	(P443)	178
linear scanner	(P1051)	1188
linguistic analysis	(P291)	1358
linguistic rules	(P280)	826
liquid crystal display	(P612)	1300
liquid crystal displays	(P833)	127
Lisp	(P956)	1265
lithography	(P412)	3
loan office description	(P285)	1066
logic	(P393)	867
logic and databases	(P938)	432

Keyword	Proj.No.	Page
logic programming	(P415)	721
	(P415)	750
	(P415)	771
	(P956)	1265
logical programming	(P973)	392
logical specifications	(P956)	1265
logistics modelling	(P932)	1671
LOKI	(P892)	402
longest processing time	(P809)	1619
LOTOS	(P410)	543
low bit rate transmission	(P925)	1161
low level image analysis	(P28)	1389
LSI applications	(P1128)	113
LV-ROM	(P901)	1124
M68020 coprocessor	(P415)	750
M68020 symbolic programming	(P956)	1265
machine learning	(P1063)	630
MACRO techniques in CIM systems	(P409)	1608
Magiscan	(P278)	1707
mail servers	(P395)	1459
maintainability	(P1609)	426
maintenance KBMS	(P892)	402
man machine interface	(P956)	1251
MANDALA	(P530)	593
manufacturing applications	(P955)	1533
Manufacturing Automation Protocol (MAP)	(P955)	1533
manufacturing cell	(P812)	1765
manufacturing data dictionary	(P477)	1648
manufacturing database	(P418)	1662
manufacturing profile	(P477)	1648
manufacturing resources planning	(P932)	1671
manufacturing systems	(P623)	1716
MARGRET	(P857)	973
MARI sensor	(P278)	1707
Markov model	(P1609)	426
MARS project	(P998)	1517
materials handling	(P812)	1765
mathematical modeling	(P1062)	1594
matrix coprocessor	(P179)	1693
matrix scanner	(P1051)	1188
matrix solving	(P179)	1693
maximal average workload	(P809)	1619
MBE	(P1270)	159
	(P1270)	159
MCAI	(P125)	503
median filter	(P97)	218
meeting aids	(P1057)	1149
megabit static RAM	(P824)	42
memory cell design	(P281)	11
memory management	(P415)	750
memory minimization	(P97)	207
menu	(P802)	287
merge analysis	(P295)	1325
MESFETs	(P843)	24
	(P843)	29
message channel	(P1057)	1149
message passing	(P937)	311
	(P1033)	566

Keyword	Proj.No.	Page
messages	(P125)	361
meta programming	(P530)	593
Meta-IV	(P496)	108
meta-knowledge	(P96)	891
metal cutting operators	(P812)	1765
metal silicides	(P1270)	159
metalorganic chemical vapor deposition(MOCVD)	(P971)	103
metaviseur	(P82)	1226
METFAC	(P1609)	426
methods	(P125)	361
metrics	(P300)	415
metropolitan area network (MAN)	(P169)	1477
mezzanine gate array	(P245)	55
MHS X400	(P33)	1822
Michalski	(P1063)	630
micro FETs	(P1128)	113
micro programmable symbolic coprocessor	(P956)	1265
micro programming	(P956)	1265
microprocessor	(P824)	42
migration server	(P878)	1111
MIKIC	(P387)	909
	(P387)	936
MIMD	(P967)	798
MIMD architecture	(P1085)	583
Miranda	(P956)	1251
MIS standards	(P33)	1822
MISOP	(P125)	503
mixed technology	(P245)	55
MMI	(P385)	1019
model based reasoning	(P932)	1671
model driven knowledge acquisition	(P1098)	665
model theoretic semantics	(P415)	771
modeling	(P384)	1638
	(P1030)	1053
	(P1062)	1594
	(P1098)	665
	(P1609)	426
modeling of office systems	(P285)	1077
modeling techniques	(P322)	1570
modularity	(P315)	451
module generation	(P97)	207
module generator environment	(P97)	218
molecular beam epitaxy	(P1270)	159
molecular beam epitaxy (MBE)	(P971)	103
molecular materials	(P443)	178
MONARCH IT440 PABX	(P387)	936
monitor	(P612)	1313
monitors	(P82)	1226
MOPS	(P82)	1205
morphological analysis	(P64)	1175
MOS modeling	(P415)	736
mosaic screen	(P901)	1124
MoSel security development	(P998)	1517
motor analysis	(P940)	850
mover transport control	(P477)	1648
moving pictures	(P925)	1161
MTA	(P718)	1816
multi media documents	(P28)	1389
	(P295)	1325

Keyword	Proj.No.	Page
multi point communication	(P1057)	1149
multi processor architectures	(P169)	1477
multi scanner	(P295)	1325
multi service network	(P169)	1477
multi-level substrate	(P958)	82
multi-processor architecture	(P97)	207
multicast communications system	(P818)	790
multichamber etch	(P574)	72
MULTILOG	(P530)	593
multimedia documents	(P395)	1459
multimedia workstation	(P395)	1459
multimodal MMI/HCI	(P385)	1019
multiple access protocol	(P169)	1477
multiplexing	(P1057)	1149
multipliers	(P843)	24
MULTOS	(P28)	1389
mutual consistency	(P285)	1066
N-well CMOS	(P412)	3
name server	(P719)	1838
name services	(P395)	1472
naming architecture	(P395)	1472
natural language	(P82)	1205
	(P280)	826
	(P311)	874
	(P967)	798
natural language processing	(P1106)	611
NC application techniques	(P409)	1608
NC machines	(P1199)	1795
network administration	(P33)	1822
network management	(P33)	1822
	(P169)	1477
	(P395)	1472
	(P718)	1816
network services	(P718)	1807
network topology	(P1085)	583
non-determinism	(P1033)	566
non-linear filtering	(P97)	218
non-linear organic materials	(P443)	178
non-linear organic optics	(P443)	178
non-monotonic reasoning	(P387)	936
non-planar 3D MOSFET	(P962)	299
Notifier (Sun)	(P1277)	325
npn-transistor	(P243)	19
NRS Lockup protocol	(P395)	1472
numeric accuracy	(P1072)	521
	(P1072)	535
numerical analysis	(P962)	299
numerical control (NC)	(P409)	1608
numerical modeling	(P612)	1300
numerical simulation	(P1128)	113
object management system	(P59)	1425
object model	(P956)	1251
object orientation	(P834)	1490
	(P956)	1276
Object Oriented Design (OOD)	(P496)	108
object sharing	(P818)	790
object-oriented architecture	(P956)	1251

Keyword	Proj.No.	Page
object-oriented database	(P125)	361
object-oriented databases	(P834)	1490
object-oriented languages	(P59)	1425
object-oriented model	(P82)	1205
object-oriented programming	(P384)	1638
	(P932)	1671
	(P973)	392
objects	(P125)	361
OCCAM	(P415)	736
	(P415)	771
	(P967)	798
	(P1085)	583
occur-check	(P415)	721
ODA	(P121)	1367
	(P295)	1325
	(P395)	1459
	(P1024)	1378
odd-even-merger	(P97)	218
ODIF	(P121)	1367
	(P295)	1325
ODIF standard	(P121)	1367
off-line programming	(P623)	1716
office assistant	(P82)	1205
	(P82)	1226
office data dictionaries	(P813)	1091
office description	(P285)	1066
office descriptive model	(P285)	1077
Office document architecture IF (ODIF)	(P1024)	1378
office document interchange formats	(P121)	1367
office document management	(P395)	1459
office documentation architecture	(P121)	1367
office environment	(P64)	1175
office formalism	(P82)	1226
office information servers (OIS)	(P231)	1413
office information systems	(P59)	1425
	(P813)	1091
	(P834)	1490
office knowledge	(P82)	1226
office models	(P82)	1226
office planning	(P878)	1111
office process migration	(P878)	1111
office requirements analysis	(P56)	1041
office support systems	(P285)	1066
	(P285)	1077
office systems	(P82)	1205
	(P82)	1226
	(P385)	1452
	(P395)	1459
	(P1030)	1053
office systems architecture	(P813)	1091
office systems LAN	(P169)	1477
office tasks	(P82)	1226
office worker	(P878)	1111
OLGA language	(P956)	1265
ontological models	(P820)	985
OOD	(P125)	361
	(P496)	108
	(P834)	1490
OOD office systems	(P834)	1501

Keyword	Proj.No.	Page
OOP	(P973)	392
OOPS	(P1117)	688
OPAMP	(P802)	287
open distributed systems	(P410)	543
open frameworks	(P1058)	251
open systems	(P718)	1807
open systems architecture	(P818)	790
open systems for CIM	(P688)	1777
Open Systems Interconnection (OSI)	(P955)	1533
operations planning	(P409)	1608
operative knowledge	(P1542)	1005
operator modeling	(P857)	811
OPS5	(P477)	1648
optical communications	(P169)	1477
optical materials	(P443)	178
optimisation of document layout	(P367)	1239
optimised production techniques	(P932)	1671
optimised production technology	(P1199)	1795
optimizer	(P802)	287
ORACLE	(P612)	1288
organisational factors	(P1030)	1053
Organization model	(P82)	1205
OSI	(P395)	1472
	(P688)	1777
	(P718)	1807
	(P812)	1765
	(P1024)	1378
OSI protocols	(P410)	543
OSI-PC	(P718)	1807
OSSAD	(P285)	1066
	(P813)	1091
OSSAD methodology	(P285)	1077
P 26	(P26)	836
P 28	(P28)	1389
P 33	(P33)	1822
P 56	(P56)	1041
P 59	(P59)	1425
P 64	(P64)	1175
P 82	(P82)	1205
	(P82)	1226
P 96	(P96)	891
P 97	(P97)	207
	(P97)	218
	(P97)	241
P 121	(P121)	1367
P 125	(P125)	361
	(P125)	503
P 169	(P169)	1477
P 179	(P179)	1693
P 231	(P231)	1413
P 234	(P234)	1101
P 243	(P243)	19
P 245	(P245)	55
P 271	(P271)	271
P 278	(P278)	1707
P 280	(P280)	826
	(P280)	955
	(P280)	963

Keyword	Proj. No.	Page
P 281	(P281)	11
P 285	(P285)	1066
	(P285)	1077
P 291	(P291)	1358
P 295	(P295)	1325
	(P295)	1341
P 300	(P300)	415
P 311	(P311)	874
P 315	(P315)	451
	(P315)	466
P 322	(P322)	1547
	(P322)	1570
P 334	(P334)	164
P 367	(P367)	1239
P 384	(P384)	1638
P 385	(P385)	1019
	(P385)	1443
	(P385)	1452
P 387	(P387)	909
	(P387)	936
P 390	(P390)	491
P 393	(P393)	867
P 395	(P395)	1459
	(P395)	1472
P 401	(P401)	375
P 409	(P409)	1608
P 410	(P410)	543
	(P410)	543
P 412	(P412)	3
P 415	(P415)	701
	(P415)	721
	(P415)	736
	(P415)	750
	(P415)	771
P 418	(P418)	1662
P 432	(P432)	480
P 443	(P443)	178
P 477	(P477)	1648
P 491	(P491)	148
P 496	(P496)	1583
P 530	(P530)	593
P 554	(P554)	33
P 563	(P563)	1137
P 574	(P574)	72
P 612	(P612)	1288
	(P612)	1288
	(P612)	1300
	(P612)	1313
P 623	(P623)	1716
P 688	(P688)	1777
P 718	(P718)	1807
	(P718)	1816
P 719	(P719)	1838
P 802	(P802)	287
P 809	(P809)	1619
P 812	(P812)	1765
P 813	(P813)	1091
P 818	(P818)	790
P 820	(P820)	985

Keyword	Proj.No.	Page
P 824	(P824)	42
P 833	(P833)	127
P 834	(P834)	1490
	(P834)	1501
P 843	(P843)	24
	(P843)	29
P 857	(P857)	811
	(P857)	819
	(P857)	973
P 878	(P878)	1111
P 881	(P881)	559
P 888	(P888)	279
P 892	(P892)	402
P 901	(P901)	1124
P 909	(P909)	1745
P 925	(P925)	1161
P 932	(P932)	1671
P 937	(P937)	311
P 938	(P938)	432
P 940	(P940)	850
P 955	(P955)	1533
P 956	(P956)	1251
	(P956)	1265
	(P956)	1276
P 958	(P958)	82
P 962	(P962)	299
P 967	(P967)	798
P 971	(P971)	103
P 973	(P973)	392
P 974	(P974)	329
P 991	(P991)	195
P 998	(P998)	1517
P1024	(P1024)	1378
P1030	(P1030)	1053
P1033	(P1033)	566
P1051	(P1051)	1188
P1057	(P1057)	1149
P1058	(P1058)	251
P1062	(P1062)	1594
P1063	(P1063)	630
P1072	(P1072)	521
	(P1072)	535
P1085	(P1085)	583
P1098	(P1098)	665
P1106	(P1106)	611
P1117	(P1117)	688
P1128	(P1128)	113
P1133	(P1133)	640
P1199	(P1199)	1795
P1262	(P1262)	352
P1270	(P1270)	159
P1277	(P1277)	325
	(P1277)	334
P1542	(P1542)	1005
P1609	(P1609)	426
PAC	(P477)	1648
FACT	(P1262)	352
FACT- PCTE	(P1262)	352
PAD	(P718)	1807

Keyword	Proj.No.	Page
panel display	(P878)	1111
paper interface	(P295)	1325
	(P295)	1341
paper role	(P295)	1325
Papillon Prototype	(P496)	108
parallel architectures	(P415)	701
parallel associative architecture	(P967)	798
parallel computations	(P1033)	566
parallel inference architecture	(P415)	750
parallel programming	(P956)	1265
parallel reduction machine	(P415)	701
parallelism	(P1085)	583
parameterised specifications	(P315)	451
parametric modeling	(P322)	1547
parametric models	(P322)	1570
parasitic capacitance	(P243)	19
PARLOG	(P956)	1251
	(P956)	1265
parsers	(P26)	836
	(P393)	867
partial evaluation techniques	(P530)	593
Pascal-SC	(P1072)	535
PCM filter synthesis	(P97)	207
PCTE	(P834)	1490
	(P974)	329
	(P1262)	352
	(P1277)	325
	(P1277)	334
	(P1609)	426
PCTE porting	(P1277)	325
PCTE-OMS	(P834)	1490
PDDI	(P322)	1547
PECUD	(P843)	29
perception	(P56)	1041
	(P612)	1288
performance	(P1609)	426
performance interpretation	(P280)	955
performance management	(P415)	750
performance metrication	(P1277)	334
period batch control	(P1199)	1795
persistent objects	(P834)	1490
persistent programming languages	(P834)	1490
Petri nets	(P56)	1041
	(P125)	503
	(P285)	1066
	(P410)	543
	(P477)	1648
	(P932)	1671
PHIGS standard	(P973)	392
phonemes	(P64)	1175
	(P291)	1358
phonetic alphabet	(P291)	1358
phonetic context	(P64)	1175
photo-lithography	(P243)	19
photodissociation of SiH ₄	(P1051)	1188
photolithography	(P1051)	1188
photoluminescence	(P1270)	159
photovideotex	(P563)	1137
physical file format	(P322)	1547

Keyword	Proj.No.	Page
PICA	(P563)	1137
pictorial images	(P28)	1389
picture coding	(P563)	1137
	(P925)	1161
picture storage	(P901)	1124
picture transmission	(P925)	1161
piezo resistive sensor	(P278)	1707
Pilot ODA	(P1024)	1378
placement	(P802)	287
	(P991)	195
plan recognition	(P280)	955
planning	(P384)	1638
	(P623)	1716
planning panels	(P878)	1111
planning systems	(P932)	1671
plant automation	(P179)	1693
plant pathology ES	(P1063)	630
plasma CVD	(P334)	164
plasma deposition	(P334)	164
plasma displays	(P612)	1300
plasma enhanced chemical vapor deposition	(P971)	103
plasma etching	(P574)	72
point spread function	(P612)	1288
	(P612)	1300
	(P612)	1313
poisson solver	(P962)	299
poly tetra fluoro ethylene	(P958)	82
poly-si doping	(P491)	148
poly-si thin film	(P491)	148
polycrystalline silicon	(P833)	127
polycrystalline TFT's	(P1051)	1188
polyimide	(P958)	82
polysilicon	(P412)	3
polysilicon deposition	(P833)	127
polysilicon emitter	(P243)	19
POOL	(P415)	736
POOL-T	(P834)	1490
portability	(P818)	790
portability of PCTE	(P1277)	334
power load control	(P387)	909
power plant control KBS	(P820)	985
PRADOS	(P892)	402
predicate transition net	(P1133)	640
prediction	(P1609)	426
predictive coding	(P563)	1137
printer server	(P367)	1239
printing devices	(P295)	1325
priority agendas	(P82)	1226
privacy	(P998)	1517
private management domain	(P718)	1816
probabilistic behavior	(P285)	1066
probing algorithms	(P271)	271
Procedure assistance	(P82)	1205
process	(P554)	33
process control	(P820)	985
	(P857)	819
process graphs	(P1033)	566
process model	(P322)	1570
	(P384)	1638

Keyword	Proj.No.	Page
process monitoring system	(P857)	973
process operators	(P1033)	566
process planning	(P409)	1608
process simulation	(P281)	11
product layer	(P96)	891
product model	(P322)	1570
	(P384)	1638
	(P998)	1517
production activity control (PAC)	(P477)	1648
production cells	(P623)	1716
production compilation network	(P1133)	640
production islands	(P1199)	1795
production planning	(P496)	108
	(P1199)	1795
production rule language	(P1133)	640
production scheduling	(P418)	1662
program proof	(P1033)	566
program specification	(P390)	491
program transform formation	(P390)	491
progressive recursive binary nesting	(P563)	1137
PROLOG	(P530)	593
	(P973)	392
PROLOG III	(P967)	798
	(P1106)	611
PROLOG on transputer	(P1085)	583
PROLOG prototyping	(P892)	402
PROLOG with numerics	(P1106)	611
PROLOG/GKS binding	(P393)	867
propositional calculus	(P1106)	611
propositional temporal logic	(P937)	311
prosody generation	(P64)	1175
PROSPECTRA	(P390)	491
protocol editor	(P1098)	665
prototyping on the PCTE	(P1262)	352
PTFE	(P958)	82
pure clause	(P415)	721
PWS-X workstation	(P395)	1459
quality assurance	(P300)	415
quantum wells	(P1270)	159
query languages	(P59)	1425
Quick Response Expert System	(P857)	973
Quick Response Expert System (QRES)	(P857)	811
Quinlan	(P1063)	630
QUINTUS PROLOG	(P28)	1389
	(P59)	1425
radiometric properties	(P612)	1313
RAISE	(P315)	466
RAISE specification language	(P315)	466
RAM	(P281)	11
rank order filter	(P97)	218
rapid prototyping	(P813)	1091
raster graphics	(P1024)	1378
Rdb/VMS	(P477)	1648
RDBMS	(P311)	874
	(P530)	593
	(P938)	432
	(P1133)	640

Keyword	Proj.No.	Page
re-use	(P974)	329
reactive ion etching	(P574)	72
real time communications	(P1057)	1149
real time control	(P179)	1693
real time expert system	(P809)	1619
real time processing	(P97)	218
	(P809)	1619
real time scheduler	(P809)	1619
real time Unix V	(P818)	790
real world testing	(P878)	1111
realtime embedded systems	(P937)	311
reasoning	(P1106)	611
reasoning engine	(P96)	891
reasoning shell	(P96)	891
recognition rate	(P295)	1341
reconfigurable transputer	(P1085)	583
reconfiguration	(P824)	42
recursive object recognition	(P28)	1389
REDIFLOW	(P967)	798
reduction	(P415)	721
reduction model	(P415)	701
reference models	(P322)	1547
reflected image	(P612)	1313
reflection images	(P612)	1313
refractory metallization	(P971)	103
register transfer description	(P97)	207
relational algebra	(P530)	593
relational data model	(P938)	432
relational database management system (RDBMS)	(P938)	432
relational graphs	(P28)	1389
reliability	(P300)	415
	(P1609)	426
reliability analysis	(P1062)	1594
reliability assessment	(P1062)	1594
remote operation servers	(P718)	1816
remote operation services	(P33)	1822
representation description tools	(P496)	108
representational redundancy	(P415)	721
requirements analysis	(P231)	1413
Research Open System for Europe (ROSE)	(P33)	1822
retinal processing	(P612)	1288
rewriting rules	(P415)	771
RF on-wafer testing	(P971)	103
RISC architecture	(P956)	1251
rise and decay times	(P612)	1300
robot cell programming	(P278)	1707
robot manipulator	(P940)	850
robot systems	(P278)	1707
	(P623)	1716
robot vision control system	(P278)	1707
robotics	(P384)	1638
	(P623)	1716
ROMULUS	(P623)	1716
ROSACE	(P974)	329
ROSE project	(P719)	1838
routing	(P802)	287
	(P991)	195
RP3	(P967)	798
RS-232	(P1024)	1378

Keyword	Proj.No.	Page
RSL	(P315)	466
RSL-RAISE specification language	(P315)	451
RSlogic	(P401)	375
Rubinoff	(P280)	826
rule based mapping assistants	(P892)	402
rule based selection	(P477)	1648
rule based systems	(P311)	874
	(P387)	936
	(P809)	1619
	(P820)	985
	(P1133)	640
rule learning	(P387)	936
rules	(P1542)	1005
safety	(P1609)	426
SAGFETs	(P843)	29
satellite control KBS	(P820)	985
SC-network	(P802)	287
scanning electron microscope	(P271)	271
scene acquisition	(P940)	850
schedule improvement	(P418)	1662
scheduler selector prototype	(P477)	1648
scheduling	(P418)	1662
	(P477)	1648
	(P496)	108
	(P809)	1619
scheduling strategy	(P809)	1619
Schottky barriers	(P1270)	159
SCHUSS filter	(P415)	750
scientific computation	(P1072)	521
script recognition	(P295)	1341
SCYLA compiler	(P64)	1175
sea of gates	(P824)	42
	(P991)	195
search systems	(P387)	909
security	(P834)	1501
	(P998)	1517
security model	(P998)	1517
seed windows	(P245)	55
seeded ZMR	(P245)	55
SEG	(P245)	55
segment concatenation	(P64)	1175
SEGRAS	(P125)	503
selective epitaxial growth (SEG)	(P245)	55
self aligned GaAs FETs	(P843)	29
self aligned technology	(P843)	24
self alignment	(P971)	103
semantic complexity	(P1542)	1005
semantic grammars	(P26)	836
semantic representation	(P393)	867
semantic type declarations	(P956)	1276
semantics	(P125)	361
semi-insulation	(P1128)	113
semiconductor simulation	(P962)	299
sensor based robot system	(P278)	1707
sensors	(P278)	1707
sentence decomposition	(P64)	1175
sentential anaphora	(P393)	867
sequence matrices	(P291)	1358

Keyword	Proj.No.	Page
sequential PARLOG machine (SPM)	(P956)	1251
serial multiplier	(P843)	24
service integration	(P169)	1477
session layer services	(P718)	1807
SESTA project	(P718)	1807
shortest processing time	(P809)	1619
SIBEMOL	(P974)	329
signal modeling	(P415)	736
signal processing	(P26)	836
SILAGE	(P97)	207
SILGO environment	(P97)	241
silicides	(P554)	33
	(P1270)	159
silicon	(P554)	33
silicon compilation	(P97)	207
	(P97)	241
	(P991)	195
	(P1058)	251
silicon compilers	(P179)	1693
silicon deposition	(P491)	148
silicon sensors	(P1051)	1188
silicon/silica interface	(P491)	148
SIMD architecture	(P824)	42
simple inheritance	(P956)	1276
simulation	(P234)	1101
	(P271)	271
	(P415)	736
	(P477)	1648
	(P612)	1300
	(P612)	1313
	(P623)	1716
	(P962)	299
	(P1058)	251
simulators	(P802)	287
situation assessment	(P820)	985
skeleton cell	(P802)	287
SKI-combinator	(P415)	701
SKILL language	(P97)	241
SLOCOP	(P1058)	251
small medium enterprises	(P909)	1745
smart power	(P245)	55
SMEs	(P909)	1745
SOAR system	(P96)	891
social and organisational perspective	(P878)	1111
soft macros	(P991)	195
software architectures	(P1024)	1378
software components	(P974)	329
software correctness	(P125)	503
software development	(P315)	451
	(P390)	491
	(P401)	375
software engineering	(P315)	451
	(P938)	432
	(P938)	432
software engineering databases	(P125)	361
software engineering environments	(P125)	361
software engineering process model	(P300)	415
software environments	(P410)	543
	(P1262)	352

Keyword	Proj.No.	Page
	(P1277)	334
software factory	(P1262)	352
software library	(P1062)	1594
software quality	(P300)	415
software reliability	(P300)	415
software toolkit	(P909)	1745
software tools	(P125)	361
	(P1262)	352
SOI-CMOS	(P245)	55
solid model exchange	(P322)	1547
solid modeling	(P322)	1547
sorted logic	(P393)	867
SPAG	(P121)	1367
	(P1024)	1378
SPAG services	(P955)	1533
SPAG/GUS profiles	(P33)	1822
Spanish linguistic analysis	(P291)	1358
specification conformance	(P410)	543
specification language	(P315)	466
specification object properties	(P125)	503
specifications	(P125)	361
	(P315)	451
	(P315)	466
	(P410)	543
SPECTRE	(P554)	33
speculative parallel evaluation	(P415)	701
speech	(P385)	1452
speech interface	(P64)	1175
	(P385)	1452
speech output	(P385)	1452
speech parsers	(P26)	836
speech recognition	(P26)	836
	(P385)	1452
speech rules compiler	(P64)	1175
speech synthesis	(P64)	1175
speech understanding	(P26)	836
	(P967)	798
SPICE circuit language	(P881)	559
SPM	(P956)	1265
spreadsheet	(P315)	466
sputtering	(P843)	29
SQL	(P311)	874
	(P1117)	688
stable run criteria	(P809)	1619
standardisation	(P322)	1547
state encoding	(P991)	195
statistics help systems	(P901)	1124
STEP	(P322)	1547
stereo matching	(P940)	850
stereo reconstruction	(P940)	850
stick editor	(P97)	218
still picture TV	(P1057)	1149
still pictures	(P925)	1161
stochastic Petri nets	(P285)	1066
strategies	(P280)	963
stream architectures	(P1033)	566
stroke matching	(P295)	1341
structure editing	(P121)	1367
subjective testing	(P563)	1137

Keyword	Proj.No.	Page
subsumption	(P415)	721
Sun	(P1277)	325
SUNCORE	(P28)	1389
supervision and control	(P857)	811
	(P857)	819
supervisory control	(P809)	1619
SUSY	(P26)	836
sweeping objects	(P322)	1547
switch maintenance	(P387)	936
symbolic dorsal architecture	(P967)	798
symbolic network language	(P881)	559
symbolic processing	(P956)	1265
SYMNET	(P881)	559
syntactic analysis	(P59)	1425
syntax directed editing	(P956)	1276
SYNTAX scanner	(P956)	1265
system architecture	(P1609)	426
system configuration	(P974)	329
system development methodology	(P998)	1517
system integration	(P812)	1765
system integration aspects	(P409)	1608
system methodology	(P285)	1077
system metrics	(P1609)	426
system modeling language	(P892)	402
stystolic array	(P824)	42
T410	(P1024)	1378
T800 transputer	(P1085)	583
tactile sensing	(P278)	1707
tape automated bonding (TAB)	(P958)	82
task analysis	(P385)	1443
TAXIS	(P892)	402
TDL	(P892)	402
technical and office protocols (TOP)	(P955)	1533
technical slope model	(P322)	1570
technology mapping	(P991)	195
technology model	(P322)	1570
telecommunications	(P387)	936
teleconferencing	(P1057)	1149
teleworking	(P1030)	1053
telewriter	(P1057)	1149
temporal logic	(P937)	311
term rewriting systems	(P125)	503
test bed simulation	(P477)	1648
test pattern generation	(P271)	271
testing	(P97)	218
text corpora	(P291)	1358
text dictation	(P291)	1358
text generation	(P280)	826
text interface	(P385)	1452
text retrieval	(P59)	1425
text-to-speech	(P291)	1358
text-to-speech interface	(P64)	1175
TFTs	(P833)	127
theorem prover	(P415)	721
theory processor	(P530)	593
thermal analysis	(P1062)	1594
thermal chip management	(P958)	82
thermal management	(P1062)	1594

Keyword	Proj.No.	Page
thermal stresses	(P1128)	113
thin film technology	(P1051)	1188
thin film transistors	(P491)	148
	(P833)	127
thin layer sputtering	(P334)	164
third wave office systems	(P878)	1111
THORN	(P719)	1838
THORN project	(P395)	1472
threat analysis	(P998)	1517
TI Explorer	(P857)	819
timed Petri nets	(P285)	1066
timing verification	(P1058)	251
TMS	(P387)	909
TODOS	(P813)	1091
token bus	(P955)	1533
tomato pathology ES	(P1063)	630
tomography	(P1128)	113
tool integration	(P1058)	251
TOOLKIT	(P909)	1745
toolkit for KBS	(P820)	985
tools communication	(P432)	480
tools for distributed systems	(P834)	1501
topology of domain actors	(P1063)	630
training for IT	(P385)	1019
transform coding	(P563)	1137
transformation systems	(P311)	874
transformational development	(P390)	491
transistor arrays	(P833)	127
transistor characteristics	(P554)	33
transistor modelling	(P971)	103
transistors	(P1270)	159
transition network compiler	(P857)	819
transitive closures	(P311)	874
transputer	(P415)	736
	(P967)	798
	(P1085)	583
tree construction	(P956)	1265
tri-layer interconnect	(P243)	19
TRIADE 60 processor	(P612)	1313
trinocular configuration	(P940)	850
triple X services	(P718)	1807
truth maintenance system (TMS)	(P387)	936
tungsten	(P554)	33
tungsten gate	(P843)	29
TV	(P1057)	1149
type checking	(P956)	1276
type model	(P834)	1490
type parameters	(P956)	1276
type state checking	(P956)	1276
type trees	(P956)	1276
typed attributed graphs	(P432)	480
typed object-oriented language	(P956)	1276
ultrafast logic	(P971)	103
uncertainty	(P59)	1425
unification	(P82)	1226
	(P415)	721
	(P1106)	611
unification grammar	(P393)	867

Keyword	Proj.No.	Page
Unix	(P530)	593
Unix mail	(P280)	826
	(P280)	955
Unix V	(P33)	1822
usability	(P59)	1425
	(P385)	1443
usability of systems	(P385)	1019
user definable application tech. for CAD/CAP	(P409)	1608
user interaction	(P833)	127
user interface	(P59)	1425
	(P82)	1205
	(P234)	1101
	(P802)	287
	(P956)	1251
	(P1058)	251
	(P1117)	688
	(P1277)	325
user interface management systems	(P956)	1251
user management tools	(P834)	1501
user models	(P280)	955
	(P280)	963
	(P857)	811
	(P857)	819
user needs in manufacturing	(P955)	1533
utterance	(P280)	826
vacuum deposition	(P1270)	159
validation	(P125)	503
	(P234)	1101
	(P998)	1517
	(P1098)	665
vanishing points	(P940)	850
VAX	(P1277)	325
VAX station 2000	(P1277)	325
VDA-FS standard	(P322)	1547
VDM	(P315)	451
	(P315)	466
	(P496)	108
verification	(P125)	361
	(P125)	503
	(P285)	1066
	(P878)	1111
	(P998)	1517
	(P1058)	251
	(P1098)	665
	(P1117)	688
verification control	(P300)	415
vias	(P958)	82
video systems	(P901)	1124
videoconferencing	(P925)	1161
videodisc storage	(P901)	1124
videotex	(P563)	1137
Vienna Development Method (VDM)	(P496)	108
viewing model	(P322)	1570
viewpoints	(P56)	1041
	(P432)	480
virtual inference machine	(P415)	750
virtual object memory	(P834)	1490
virtual terminal	(P718)	1807

Keyword	Proj.No.	Page
vision	(P612)	1288
vision modeling	(P612)	1313
vision models	(P612)	1288
visual perception	(P612)	1288
visual sensing	(P278)	1707
visual system technology	(P612)	1313
VLSI	(P97)	218
	(P179)	1693
	(P243)	19
	(P278)	1707
	(P415)	736
	(P888)	279
	(P956)	1251
	(P958)	82
	(P967)	798
	(P1058)	251
VLSI design	(P991)	195
VLSI tools	(P888)	279
voice annotation	(P385)	1452
voice retrieval system	(P901)	1124
von Neuman (non-) architectures	(P415)	750
VT DIS 9041	(P718)	1807
vulnerability	(P998)	1517
wafer processing	(P971)	103
wafer scale integration (WSI)	(P824)	42
wafer scale memory	(P824)	42
Warren abstract machine	(P415)	771
wave form acquisition	(P271)	271
wideband communication	(P169)	1477
wideband speech	(P1057)	1149
window manager	(P973)	392
windowing	(P956)	1251
wireframe models	(P322)	1547
WOMBAT	(P1058)	251
word processors	(P385)	1452
	(P1024)	1378
work planning systems	(P409)	1608
workcell	(P623)	1716
workcell controller	(P932)	1671
workpiece handling	(P278)	1707
workstation	(P82)	1226
	(P956)	1251
	(P956)	1265
X-ray lithography	(P574)	72
X-window server	(P82)	1205
X/OPEN	(P33)	1822
X25 services	(P718)	1807
	(P718)	1816
X29 services	(P719)	1838
X400	(P33)	1822
	(P395)	1459
	(P718)	1807
	(P718)	1816
	(P1024)	1378
XSQL	(P311)	874
XXX services	(P718)	1807
zone melting crystallisation	(P245)	55



ISBN Part 1: 0 444 70331 4
ISBN Part 2: 0 444 70332 2
ISBN Set : 0 444 70333 0