ICL TECHNICAL JOURNAL

Volume 5 Issue 2 November 1986

Published by
INTERNATIONAL COMPUTERS LIMITED
at
OXFORD UNIVERSITY PRESS

ICL TECHNICAL JOURNAL

The ICL Technical Journal is published twice a year by International Computers Limited at Oxford University Press.

All correspondence and papers to be considered for publication should be addressed to the Editor.

The views expressed in the papers are those of the authors and do not necessarily represent ICL policy.

1986 subscription rates: annual subscription £16 UK, £19 rest of world, Us $40 N. America; single issues £10 UK, £11 rest of world, US $25 N. America. Orders with remittances should be sent to the Journals Subscriptions Department, Oxford University Press, Walton Street, Oxford OX2 6DP, UK.

Printed by H Charlesworth & Co Ltd, Huddersfield ISSN 0142-1557

ICL
TECHNICAL
JOURNAL

Contents

Volume 5 Issue 2

Foreword **167**

The Management Into the 1990s Research Program
M.S. Scott Morton **169**

Managing strategic ideas: the role of the computer
C. Eden **173**

A study of interactive computing at top management levels
C.J. Martin **184**

A management support environment
N.C. Austin **196**

Managing change and gaining corporate commitment
P. Hall **213**

An approach to information technology planning
S.J. Pollard and C.R. Crawford **228**

Preparing the organisation for IPSE
P.W. Veasey and S.J. Pollard **253**

Global Language for Distributed Data Integration
P.M. Stocker **274**

The design of distributed secure logical machines
R.W. Jones **291**

Mathematical logic in the large practical world
E. Babb **309**

The ICL DRS 300 management graphics system
R.J. Bunyan **317**

Performance of OSLAN local area network
A. Maynard-Smith **326**

Experience with programming parallel signal-processing algorithms in Fortran 8X
A. Wilson **344**

Book review **351**

Notes on the authors **353**

Foreword

This issue of the Technical Journal has a group of papers on a theme that may be expressed as 'Information Technology as a tool of management'. The importance of management, especially of the large and complex organisations that characterise the modern industrial economy, is self evident and a glance at the business pages of any serious newspaper, any day, will provide plenty of evidence of the seriousness and difficulty of the problems it gives rise to. Management is largely about information, so Information Technology should be pressed into service as an aid when and wherever that seems profitable. And as Information Technology increases its impact on all aspects of business and economic life, the understanding of the management impact becomes a critical factor in the effective use of information systems.

The main issues are put by Professor Michael Scott Morton in his introductory paper. He is Director of the Management into the Nineties research program at the Sloan School of Management, Massachusetts Institute of Technology. ICL is one of the sponsors of this program, the others being

American Express Travel-Related Services Company
Arthur Young & Company
British Petroleum Company plc
Bell South Corporation
Digital Equipment Corporation
Eastman Kodak Company
General Motors Corporation
MCI Communications Corporation
United States Internal Revenue Service

In ICL, understanding and mastering the managerial impact of technology alongside the understanding of the technology itself is a critical element of our forward business planning. Our chief reasons for participating in this program can be summarised as follows:

1 the issues addressed by the program are key factors that will affect the company itself as a business, and equally will affect its customers
2 the program will be a valuable source of information about critical issues facing management, both now and in the future; and of possible solutions that can be reflected in ICL's products and services.

We look forward to publishing papers dealing with results from the MIN program.

The 'Management' papers are the first six following Professor Scott Morton's introduction. The two from UK Business Schools (Eden at Bath, Martin at Durham) reflect our increasing interest in and involvement with academic institutions of this type. Those by Pollard & Crawford and Veasey & Pollard respectively arise from some recent thinkng in ICL's Group Information Services Division, the Division which, as one of these papers states, is responsible for providing the company with the software services needed to run its own business. Veasey & Pollard deals with the particular problem of managing a large software project. Hall on Decision Conferencing describes a technique evolved in ICL for arriving at managerial decisions and gaining commitment to the decisions arrived at; and Austin discusses the problems and needs of the physical environment and describes ICL's own solution, colloquially (but officially) known as the 'pod'.

Several of the other papers have managerial relevance. Bunyan's is a short technical account of the facilities for graphical presentation and manipulation of information provided by the recently-announced ICL DRS300 networked system. Distributed systems will certainly increase in importance as technical tools of management: Stocker discusses the language for addressing these and Jones the problems of control and security of information handling in such systems, both papers making specific proposals. Babb's paper arises from work he has been doing on the use of mathematical logic in business systems.

The last paper in this issue is quite different – a review by Dr. Pinkerton of Professor Wilkes's 'Memoirs of a Computer Pioneer'. Maurice Wilkes, who built one of the world's first true computers, the Cambridge EDSAC, was a founder member of the Editorial Board of this journal and gave invaluable help and advice in its setting up; it is very pleasing to have this opportunity to show our appreciation.

J. Michael Watson
ICL Director, Marketing and Technical Strategy.

The Management Into the 1990s Research Program

Michael S. Scott Morton
Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA.

The Sloan School of Management at M.I.T. has embarked on a unique research program to explore the impact of I.T. on organizations. This five year $5 million program involves a consortium of ten sponsors and an interdisciplinary set of 15 M.I.T. faculty. The goal of the program is to identify the nature of impacts that I.T. will have on organizations and the implications of these for the organization's mission, structure, and management practices.

Program premises

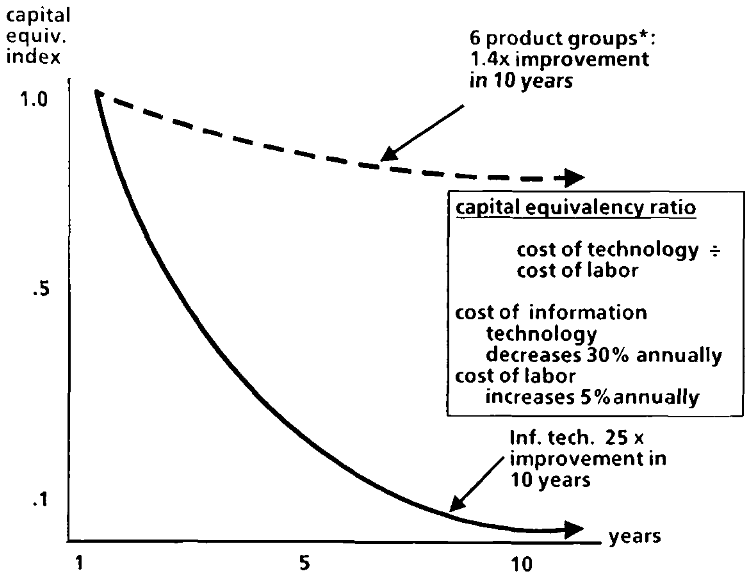
An important premise of the program is the fact that Information Technology has moved out beyond mere mainframe computers. While progressive Data Processing departments have recognised that any computer department must now include in its thinking, mainframes, mini-computers and micro-computers, there are still organizations who have failed to exploit these technologies for competitive advantage. This gap between changing computer technology and its lack of exploitation is further aggravated by the ongoing convergence between computers and telecommunications. The emerging network of differeing nodes of computer power and data bases in the hands of people with knowledge work to do is causing fundamental change in many leading organizations.

I.T., then, in the context of the 1990s program, represents the full panoply of computers, communications, robotics, professional workstations and 'chips' used to create 'intelligent' products.

All of this technological change would not justify M.I.T. and a disparate group of sponsors to explore its impact if it were not for the fact that we all view the world as being in a state of considerable turbulence and likely to remain so for some time to come. Thus, for example, we are in an era of increasing global competition that not only is creating new business opportunities, but is redefining traditional business relationships among customers and suppliers, managers and their workforce and governments and the private sector. The success of these new alliances will depend in large measure on the ability of people to adapt to change and to capitalize on often unexpected opportunities.

The program's view of the world, then, is one of considerable environmental turbulence to which the organization must adapt if it wishes to survive successfully over the decades ahead. I.T. provides the organization with one set of tools by which to adapt. While these tools have been available for some 25 years they have taken on a whole new shape by virtue of two factors. The first of these is the technology itself. Miniaturization (and the resultant large drop in cost) and the convergence with telecommunications referred to above combine to make available a quite different form of power than the previous generations of data processing.

The second factor leading to our ability to use I.T. to adjust to external turbulence is the shift in relative costs of I.T. capital and other forms of capital. There is a major economic imperative by virtue of the fact that there is a changed economic relationship between capital and labour prices caused by several generations of prodigious performance improvements in the building blocks of I.T. (memory, computation). This has not been at all matched by other industries, and is starkly evident when matched with their labour costs. Historically labour and capital goods prices have tended to move similarly over time, (see Fig. 1), whereas the I.T. capital:labour ratio has dropped dramatically (ref Benjamin/MSSM). In a typical 10 year period the latter improved 25 times while traditional industries only improved 1.4



capital equivalency ratio

cost of technology ÷ cost of labor

cost of information technology decreases 30% annually
cost of labor increases 5% annually

*source-Bureau of Labor Statistics, producer price index, compensation per hour, non farm, business sector—1950 thru 1980

Fig. 1. Capital Equivalency Ratio-Information Technology versus 6 product groups 1950–1980

times. The dramatic difference between these two curves suggests the existence of enormous potential for those organizations willing to seize the opportunity.

Conceptual structure

In order to understand how one might go about finding where the opportunities and challenges may lie and what has to be done to achieve them the Program established a conceptual structure to guide the initial research. This structure builds on the pioneering work of two separate streams of research, that of business historian Alfred Chandler and organizational behaviourist Harold Leavit. Among many other works, Chandler in his book 'Strategy and Structure' lays out the clear historical relationship between the changes in an organization's strategy and the corresponding changes in its structure. Leavit, based on interpretations of the work of many behavioural scientists, showed that an organization can usefully be viewed as consisting of four sets of forces, namely its tasks, technology, people and structure. These, he showed, exist in a state of dynamic equilibrium.

Building on their views, with some extensions, and explicitly recognising the existence of the external environment, the 1990s Program developed the model in Fig. 2 as the conceptual structure on which to base our thinking. As an illustration of its use in practice, one can take the well documented American Hospital Supply case. As distributors of medical supplies they felt the constant pressure of a myriad of competitors and increasingly low margins due to the rising labour cost of doing business in an expanding

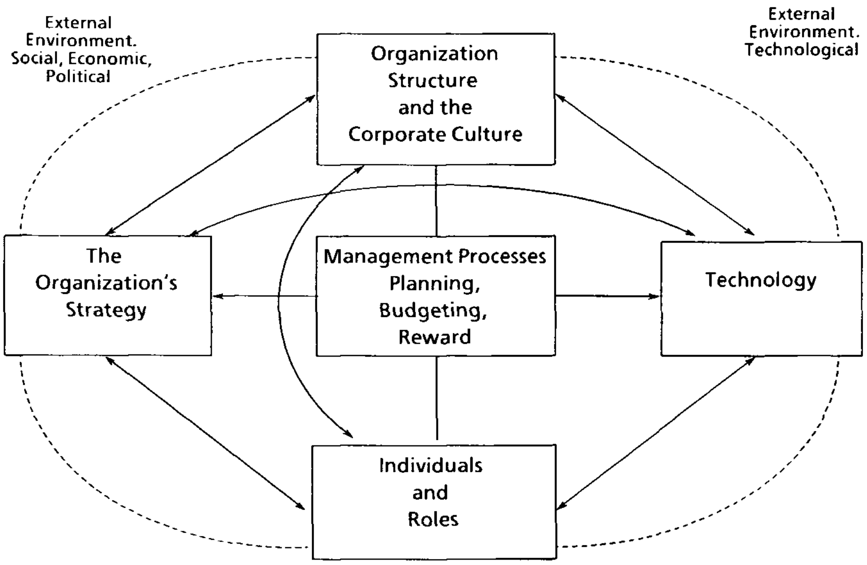

External Environment. Social, Economic, Political
External Environment. Technological
Organization Structure and the Corporate Culture
The Organization's Strategy
Management Processes Planning, Budgeting, Reward
Technology
Individuals and Roles

Fig. 2. Conceptual Structure Model for the 1990s Program

market in a labour-intensive industry. By giving their customers their own terminals for direct order entry they lowered costs and shortened lead times. They found however a major effort was required to change the skills of the people working with this new I.T. based system. In addition it was necessary to change the organization structure to focus on customer support, not just on products delivered, and to modify the salesforce incentives to recognise that the salesmen were no longer supposed to merely take orders but rather to anticipate customer needs. Success with I.T. for competitive advantage seems to involve maintaining an active balance between all these four sets of forces; ignore any, and the attempt to use I.T. on strategically important tasks tends to fail.

The initial studies in the 1990s portfolio have been concerned with the impact of I.T. on the organization's ability to adapt to external turbulence. The individual projects are concerned with the nature and management of change enabled by I.T. For example, if we expect I.T. to permit organizations to utilize more home workers, connected electronically, then this may imply a significant change in how we organize, coordinate, manage and reward such work. Each of the 90's studies will result in publications that will be available in a Working Paper format from the Sloan School at M.I.T.

As ICL and the other nine sponsors interact with the faculty and discuss the nature of our findings, this will lead to further insights and research opportunities. From this partnership we expect to focus in on some questions of central importance to management as we chart a path through these turbulent times.

Managing strategic ideas: the role of the computer

Colin Eden
School of Management, University of Bath

Abstract

This paper discusses the use of computer technology in the field of managing strategy in large organizations. This use of computers is discussed in the context of a programme designed to help small managerial teams work together on the selection of strategic options. The programme is referred to as SODA (Strategic Options and Development Analysis) and has been devised by the Strategic Decision Support Research Unit at the University of Bath.

Background

For several years the author and other co-workers have been researching the field of problem solving in teams (for example Eden, Jones & Sims, 1983; Eden, 1985; Eden *et al* 1981). The problems of interest are those that are generally regarded by the team members as 'messy' and 'soft', those that are not obviously tractable through number crunching techniques. Initial research was undertaken by working 'live' with a variety of organizations including Bath City Council, Shell International, British Telecom, and Business Press International.

Working with groups in these organizations led to the development of what is now a well established Operational Research and Soft Systems methodology known as 'cognitive mapping' (Bennett & Huxham, 1985). Put most simply cognitive mapping, which will be illustrated later, is a graphical way of modelling the interconnection of ideas and argument that people use when they are thinking and describing their problem. The modelling system was founded on the belief that language is the common currency of organizational problem solving where 'the world is presented in a kaleidoscope flux of impressions which has to be organised by our minds – and this means largely by the linguistic systems in our minds ... we cut nature up, organize it into concepts, and ascribe significances as weights largely because we are parties to an agreement to organize it in this way' (Whorf, 1956) Theoretically it is founded in Personal Construct Theory (Kelly, 1963) a body of cognitive psychology theory that argues that man is continuously striving to 'make sense' and 'predict and control' his world, which is what we take it that we try to do with organizational problems.

Cognitive maps are networks of ideas linked by arrows. The arrows indicate the way in which one idea leads to or affects another. Thus a map is a network of nodes and links (a 'directed graph') and it seemed helpful to devise computer software that would help with storing the maps and with their analysis. This relatively simple software called 'COPE' (Eden, Smithin & Wiltshire, 1980; Eden, Smithin & Wiltshire, 1985) proved itself to be a significant aid in the developing methodology for working with teams and so became an integral part of the process of working on problems.

Continuing work with internal consulting groups gradually led to a convergence of the author's views with those of senior managers about the importance of corporate planning. The implication of this convergence was the identification of the PROCESS of strategy development in organizations, the methodology, and the computer software that had been developed for team problem solving as appropriate to the management of strategy. Also the problems that had been identified by those working on strategic management (Denning, 1984; Hussey, 1984) seemed to relate to those that had been at the core of the work with problem solving in teams. This convergence led to the author becoming involved with British Telecom in the development of strategy for privatisation, and with two other multinationals in attempts to use the approaches developed for problem solving with those facing the problems of the management of strategy.

Assumptions about strategic management

The introduction mentioned the convergence of the author's and manager's views on strategy concerning the problems facing strategists. These views are:

- there is a pressing need for converting strategic thinking into strategic action

  implies that
  any methodology must be explicitly action oriented rather than being oriented to descriptions or scenarios of the environment

- developing strategic direction is about controlling and managing the future rather than forcasting it

  implies that
  thinking about strategy should not focus on refining the accuracy of forecasting but rather should attend to the business of managing the environment and particularly competitors

- strategy fails more often because senior managers at the centre of the organization do not allow their strategies to be influenced by those who know about local conditions

implies that
strategy must be developed with the active involvement of 'boundary spanning' individuals as well as senior managers; that is, strategy can be developed through workshops that have the participation of those managers who both need to think about strategy and yet have direct 'firefighting' responsibilities

- different groups bring to the consideration of strategy different, but important, perspectives and experience

  implies that
  consideration of strategy must involve a process and modelling approach that can hold on to conflicting views and merge together different perspectives on the same topic

- different interest groups within the organization need to see different outputs from the process of developing strategy

  implies that
  the strategic data base must be amenable to being 'sliced' in a variety of ways, contingent upon the specific persons in the group, their responsibilities and role within the organization

- strategic issues addressed by different groups are interdependent

  implies that
  when addressing one specific strategic issue the interdependencies must be automatically flagged

- there is a danger of concentrating too much on a single goal as being of paramount importance; in practice every goal is qualified by others and every new strategy constrained and enhanced by a network of other goals

  implies that
  corporate goals comprise a form of hierarchical system and must be represented as such

- a small proportion of strategic data is quantitative and a large proportion is qualitative, or 'soft', information

  implies that
  ways of working on strategy must facilitate working with ideas, argument and language

- different parts of strategic management data are confidential to different groups within the organization

  implies that
  computer software that manages the database must be able to screen out data according to who is using it

Strategic (cognitive) mapping

When managers talk about their view of the future and how their organization should act strategically with respect to it they use language which is designed to argue for why the world is like it is and how it might be changed. Take a simple example of one person expressing a view about profit sharing:

> 'The latter-day Labour party, aiming to appeal upmarket, is in a more ambivalent position. At a time when they are looking for a concordat with the unions, union opposition to profit sharing (because of the fear of weakening the role of collective bargaining) is hard to avoid. American experience with Employee Share Ownership Plans since the early 1980s has led to a drop in union membership in firms with these profit sharing schemes.' [extract from the *Guardian* newspaper 'Who gains from profit sharing?' by Jane McLoughlin May 14 1986. (slightly modified)]

Figure 1 shows how this may be converted into a cognitive map. Important phrases are selected which capture the essential aspects of the arguments. An attempt to capture the meaning of each phrase is made by trying to identify the contrasting idea, thus 'Labour support for profit sharing' is contrasted with 'ambivalence towards profit sharing' and 'up-market appeal' is contrasted with 'working class appeal' (this is an attempt by the coder to understand the meaning of up-market by making a guess; see Eden *et al* 1983 for further discussion of the coding of meaning). The meaning of a phrase and its contrast (a 'concept') is further elaborated by considering the argumentation that links the concept with others (in a map this is shown by the linking

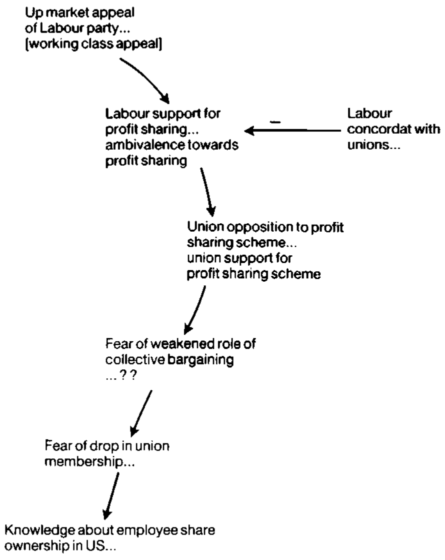

Up market appeal of Labour party... [working class appeal]
Labour support for profit sharing... ambivalence towards profit sharing
−
Labour concordat with unions...
Union opposition to profit sharing scheme... union support for profit sharing scheme
Fear of weakened role of collective bargaining ...??
Fear of drop in union membership...
Knowledge about employee share ownership in US...

Fig. 1. An example of a cognitive map

arrows). However the phrases and linking arrows are not precisely a replication of the language used by the person speaking, they have been modified to reflect the need for an 'action orientation' requirement set out above. Each of the concepts in Fig. 1 is written as a call to action to suggest an option. Similarly the argumentation (direction of the arrows) is such that an option always leads to a desired outcome, with the most important outcome hierarchically superior to others. In Fig. 1 the highest order goal is taken to be the Labour party seeking an up-market appeal and all the options are taken to have implications for appeal. In addition, the map indicates the nature of the argumentation by adding a negative sign to the end of an arrow if the first phrase of one concept relates to the second phrase of the other concept (thus 'union opposition' leads to 'ambivalence rather than support').

The map in Fig. 1 is exceedingly simple; typically a map for problem solving might contain several hundred concepts, and for strategic management maps several thousand. However, using it to identify possible strategic options reveals the principle of analysing the structure of maps. In the first instance each 'tail' (the concept at the bottom of a chain of argumentation) is identified and tested as a possible strategic intervention. Therefore 'fear of drop in union membership' and 'concordat' are tested as possible options – is there any way in which a drop in union membership can be countered? is a concordat with the unions important? (The formal tail of 'knowledge about share ownership in the US' might be ignored because it is mostly a contextual example although it could be taken as an option to 'rubbish' the knowledge, and so reduce the 'fear of a weakened role'). The first question seeks to discover further concepts which have implications for the fear about the levels of union membership (that is, elaborate the map by inserting new options/tails); and the second question tests whether the arrow between concordat and support can be erased.

Further exploration of options follows by moving up the concept hierarchy by considering ways of countering 'weakened role' and then general ways of countering 'union opposition'; that is, elaborating the map by seeking to discover further explanations for why the situation is as it is. Looking for options reveals an important role for trying to identify contrasts within concepts, for the contrast is the essence of action; for example, in considering countering 'weakened role' it is important to identify the nature of the contrast – it might be 'strengthened role' or it might be 'less weakened role', each having different implications for identifying an intervention.

Whilst Fig. 1 is a 'cognitive' map because it is a model of the *thinking* of one person, in strategic management work the map will be an aggregation of many cognitive maps and is not called a 'cognitive map' but rather a 'strategic map'. The map is typically developed through cycles of workshops involving four to six managers attending a workshop focussed on a particular strategic issue. However, the maps may also be based upon interviews or documentary material. In some cases it has been appropriate to follow the style of political scientists in their analysis of argumentation within a variety of documents expressing the strategic issues facing the organisation (Axelrod, 1976).

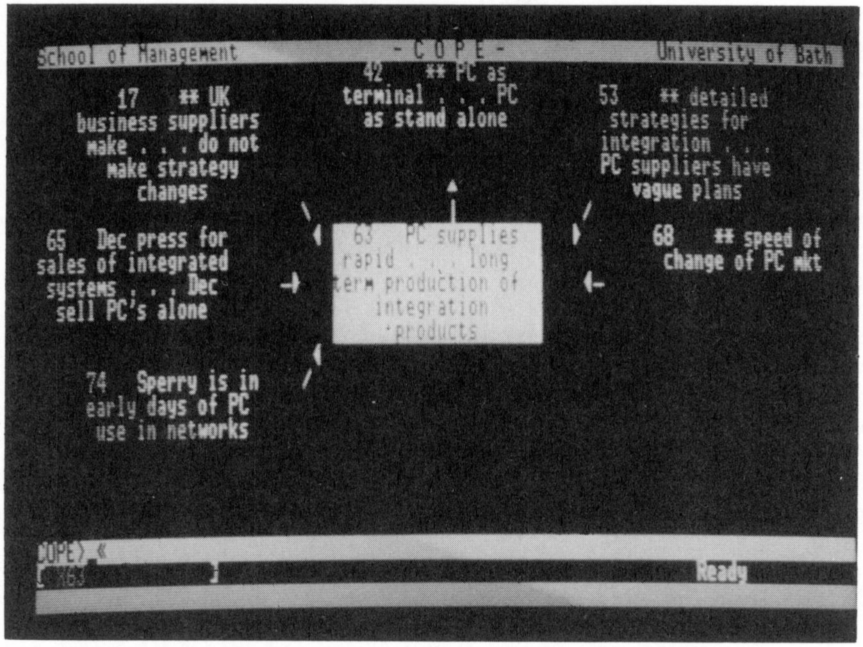
Complex data, complex structures, and the role of computers

The example in Fig. 1 contains seven concepts; strategic maps typically contain several thousand concepts covering many interdependent strategic issues. Information technology can clearly help in managing this complexity. The software that has been and currently is being redeveloped from its problem solving origins is aimed at using the power of computers to manage large quantities of qualitative data. This section presents some of the more important aspects of this software.

The software has been designed to accept two forms of input: firstly, the direct input of concepts with each concept automatically numbered and links between them entered by a simple command ('54 + 65 – 188' representing an arrow from concept 54 to 65 and also to 188 with a negative sign); and secondly through a controlled dialogue. The mapping principles make no demands for consistency or non-conflictual argumentation: a *conflicting* line of argumentation can be entered in the same way as, and alongside, the alternative view.

From the basic database of concepts and links it is possible to recover aspects of the map in a variety of ways. The most obvious facilities enable the user to search for text, list ranges of concepts, and play back parts of the argumentation. A frequently used display allows the user to place any concept in the centre of a VDU screen and see the adjacent concepts. Fig. 2 is an example of an exploration command of this nature – colours are used to differentiate concepts that possibly explain and those that are the possible consequences of the concept in the centre. In workshops that are aimed at agreeing a portfolio of strategic options this command is used to focus attention on a sequence of concepts that may be strategic interventions. As each option is displayed a group of managers are able to explore concepts at the periphery of the screen in order to gain further meaning, elaboration and consequences for action. The command clears the screen quickly and replaces it with the new concept in the middle and yet in the same position relative to the previous concept.

Figure 3 shows an alternative form of representation which simply replays some part of the argumentation itself. A command is used to either replay the consequences of an intervention or alternatively the explanations of an intervention. The amount of argumentation replayed is determined by the structure of the map – the replay stops at a branch point. Similarly a simple command (e.g. c51, 76) replays all the argumentation for the consequences of concept number 51 for concept number 76, any *conflicting views* will appear, and several consequential arguments may have a positive impact on the outcome and several may have a negative impact.



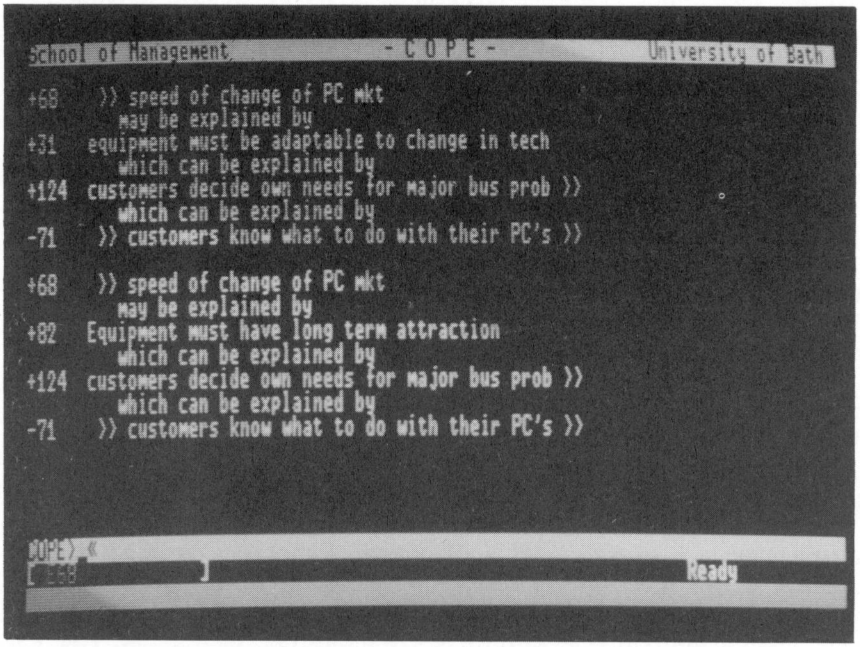
```
School of Management              - C O P E -              University of Bath
                                   42    ** PC as
        17    ** UK               terminal . . . PC         53    ** detailed
  business suppliers               as stand alone            strategies for
     make . . . do not                                      integration . . .
       make strategy                                        PC suppliers have
         changes                                               vague plans

  65   Dec press for              63   PC supplies           68    ** speed of
 sales of integrated            rapid . . . long              change of PC mkt
  systems . . . Dec           term production of
   sell PC's alone                integration
                                   products

     74   Sperry is in
     early days of PC
      use in networks

COPE> «
[ [illegible] ]                                             Ready
```

Fig. 2.



```
School of Management              - C O P E -              University of Bath
+68    >> speed of change of PC mkt
         may be explained by
+31   equipment must be adaptable to change in tech
         which can be explained by
+124  customers decide own needs for major bus prob >>
         which can be explained by
-71    >> customers know what to do with their PC's >>

+68    >> speed of change of PC mkt
         may be explained by
+82   Equipment must have long term attraction
         which can be explained by
+124  customers decide own needs for major bus prob >>
         which can be explained by
-71    >> customers know what to do with their PC's >>

COPE> «
[ [illegible] ]                                             Ready
```

Fig. 3.

While there are other forms of output they depend upon having conducted some analysis to focus upon a particular sub-system of concepts. The software currently offers a number of different ways in which a large map can be analysed. A useful approach for large maps is to attempt to *slice* the complete model into clusters. The clusters are formed through an algorithm which uses the link structure to determine the similarity of one concept to another, based upon calculating a modified Jaccard coefficient as the measure of similarity (Everitt, 1974). The overall aim is to split the model into groups which have a minimum number of inter-group links. Naturally there is a trade off between the 'exhaustiveness' of this process and its speed, and between the coarseness of the original data and the arbitrariness of pure distinctions between membership and non-membership of a cluster. Nonethe less experience shows that each cluster is likely to result in the identification of a '*strategic issue*'. Within each cluster there will be 'heads' (those concepts at the end of the chains of argument) that represent the goals to be attained within this particular strategic issue. The preferred size of a cluster can be chosen so that each issue is 'manageable' – a typical cluster will contain 25–30 concepts. Once clusters have been formed they are stored in 'groups'. Groups can be created using the clustering algorithm and by storing any other subset of concepts such as all concepts that relate to a particular division of the organization.

When a model has been structured by forming groups, any of the groups can be printed as a map. For maps of this size the operation is nontrivial and involves calculating the optimal position for each concept so as to reduce the number of crossing arrows, and to provide natural 'flow lines' along chains of concepts. The software uses a combination of 'principal coordinate analysis' (Gower, 1966) and series of complex heuristics. Principal coordinate analysis calculates the position of each concept in $(n - 1)$ dimensional space, where n is the number of concepts. The heuristics algorithm then attempts to allow for the amount of text in each concept and convert the coordinates into two dimensions laid out on A4 size sheets. The module that has been designed for this task is specific to the problems faced in managing cognitive maps, but also has many general applications for the layout of any network structure with nodes and links. The mapping facility also allows the user to redraw the draft produced by the algorithm so that any special requirements can be allowed for and the occasional error made by the heuristics can be corrected. The requirement for seeing *goals as an hierarchical network* means that it is helpful for maps to be drawn by the computer in an hierarchical fashion if desired. The maps of the strategic issues are often used as the device for guiding strategy discussion by enlarging them from A4 to A0 size. The enlarged maps are pinned to the wall and used as the focus for discussion.

A strategic database of this sort attributes equal importance to each concept in the first instance. Often a managerial group needs an overview of the major issues so that the *interdependency between issues* can be fully appreciated. The software provides two ways of establishing an overview. The first enables the clusters/groups to be given a title to describe the content of the group and each title may then be treated as a concept which is connected to others; thus the software is able to treat the network of clusters as a map in its own right so that any of the usual output facilities can be used (e.g. screen explores of interrelated clusters, a map of the clusters, etc). The second analysis permits the user to identify key concepts and 'collapse' the model into a smaller model containing only the key concepts that are related to one another according to the paths of other less important concepts that exist between them. This collapse facility also enables new models to be created which have screened out particular aspects of the overall model that should remain confidential.

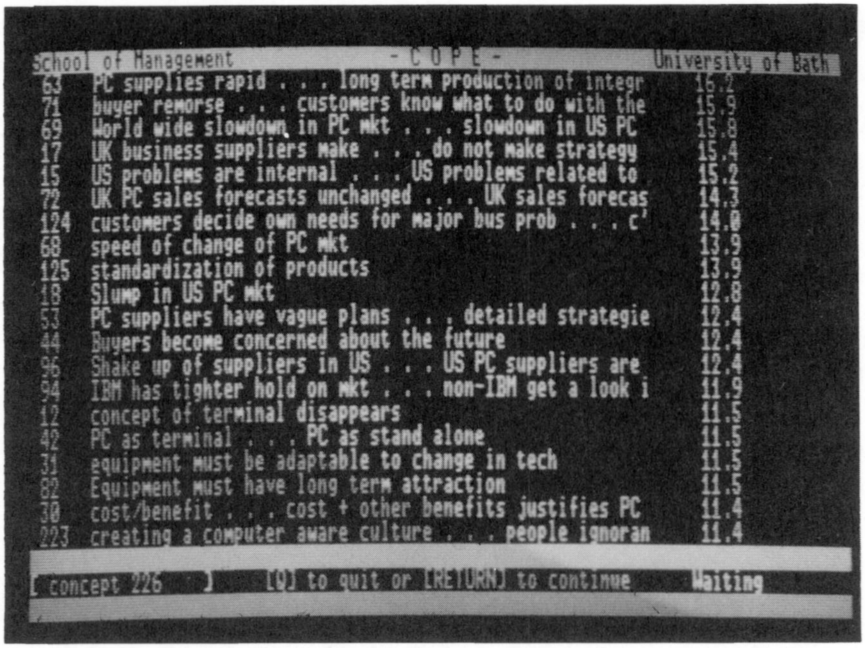
A management team often demands an analysis of the data to help them identify 'the nub of the issue'. Clearly such an analysis is always dependent upon the specific judgements that are made of the content of the data; however, analysis of the structure of the data usually provides some powerful help in identifying the core elements of issues. The methods of analysis that are available using the software focus upon a variety of different measures of the relative density of concepts in parts of the map. Thus, the software will compute the 'centrality' of each concept by i) simply calculating the number of concepts immediately related to it; ii) calculating the number of paths of argumentation that support and are consequential to the concept; iii) calculating the extent to which these paths of argumentation are well elaborated; and iv) calculating the density of concepts surrounding the concept by weighting those closest higher than those further away on an exponentially reducing basis. Inspection of each of these relative measures provides an insight into which might be 'core concepts'.

If core concepts can be established then another command within the software enables the user to print out a map of those concepts within a specified number of 'bands' from the core (see Fig. 4).

The software enables the user to define subsystems of different parts of the map. Each of eight subsystems acts as a reservoir for concepts. The subsystems are designed to enable the user to ask logic questions of the contents; thus one subsystem may contain all the data relating to 'financial' issues, another contain data that relates to 'short term' issues, another contain data that relates to the 'business systems division'. It is then possible to request an analysis that shows which short term issues are financial in nature but do not impact on the business systems division. Specifically the analysis possibilities are i) show what is not in a subsystem, ii) show what is in one subsystem and not in another, iii) show what is in two subsystems. Because the results of each test can be placed in another subsystem the logic tests can isolate highly specific needs.

The nature of research on strategic decision support

The above commentary on the role of computers in the management of complex qualitative data provides a brief insight into software that has been



```
School of Management              - C O P E -                  University of Bath
63   PC supplies rapid . . . long term production of integr      16.2
71   buyer remorse . . . customers know what to do with the      15.9
69   World wide slowdown in PC mkt . . . slowdown in US PC       15.8
17   UK business suppliers make . . . do not make strategy      15.4
15   US problems are internal . . . US problems related to       15.2
72   UK PC sales forecasts unchanged . . . UK sales forecas      14.3
124  customers decide own needs for major bus prob . . . c'      14.0
68   speed of change of PC mkt                                   13.9
125  standardization of products                                 13.9
18   Slump in US PC mkt                                          12.8
53   PC suppliers have vague plans . . . detailed strategie      12.4
44   Buyers become concerned about the future                    12.4
96   Shake up of suppliers in US . . . US PC suppliers are       12.4
94   IBM has tighter hold on mkt . . . non-IBM get a look i      11.9
12   concept of terminal disappears                              11.5
42   PC as terminal . . . PC as stand alone                      11.5
31   equipment must be adaptable to change in tech               11.5
82   Equipment must have long term attraction                    11.5
30   cost/benefit . . . cost + other benefits justifies PC       11.4
223  creating a computer aware culture . . . people ignoran      11.4

[ concept 226    ]     [Q] to quit or [RETURN] to continue         Waiting
```

Fig. 4.

designed to be highly flexible in its use. Embedded within the software is a high-level language that enables more sophisticated users to develop new analysis, output and input routines to suit their own requirements. The purpose of flexibility is to reflect the need for the software to be of service to the *process* of strategic management and so facilitate the *involvement of large numbers of people at different levels in the organization.*

This report has discussed the presumptions that have guided the process of strategy management being developed. It has also outlined the current state of the computer software that has necessarily grown to aid this process. At the present time field testing with several large organizations is guiding extensive further development. Both theory, technique and practice are continually being put to the test by working on current strategic issue of concern to senior managers (Eden, Smithin, & Williams, 1986).

The aims of the research are to develop a theory and practice of developing strategic direction which:

- is based on the wisdom and experience of managers at the top of the organization and at the 'coal-face'

- is coherent between the vision of top management and the firefighting activities of operational managers

- reduces bounded vision and group think
- changes the way in which managers interpret their task from forecasting to managing and controlling
- is action oriented
- and leads to consensus and commitment from people involved rather than compromise and control

References

1 AXELROD, R. (1976). *Structure of Decision*, University of Princeton Press: Princeton, New Jersey.
2 DENNING, B. (1984). 'Report on the Paris Conference on Making Strategy Work', Published for the Society for Long Range Planning's Newsletter.
3 EDEN, C. (1985). 'Perish the thought': *Journal of the Operational Research Society*, **36** (9), 809–819.
4 EDEN, C., JONES, S., SIMS, D., and SMITHIN, T. (1981). 'The Intersubjectivity of Issues and Issues of Intersubjectivity'. *Journal of Management Studies*, **18** (1), 37–47.
5 EDEN, C., JONES, S. and SIMS, D. (1983). '*Messing about in Problems*', Pergamon Press, Oxford.
6 EDEN, C., SMITHIN, T. and WILTSHIRE, J. (1980). 'Cognition, Simulation and Learning'. *Journal of Experiential Learning and Simulation*, **2** (2), 131–143.
7 EDEN, C., SMITHIN, T. and WILTSHIRE, J. (1985). 'COPE User Guide and Reference Manual. Bristol: Bath Software Research.
8 EDEN, C., WILLIAMS, H. and SMITHIN, T. (1986). 'Synthetic Wisdom: The design of a Mixed Mode Modelling System for Organizational Decision Making'. *Journal of the Operational Research Society*, **37** (3), 233–241.
9 EVERITT, B. (1974). '*Cluster Analysis*', John Wiley & Sons, New York.
10 GOWER, J.C. (1966). 'Some distance properties of latent root and vector methods used in multivariate analysis', *Biometrika*, **53** (3, 4), 325.
11 HUSSEY, D. (1984). 'Strategic Management. Lessons from Success and Failure', *Long Range Planning*, **17** (1).
12 HUXHAM, C. and BENNETT, P. (1985). 'Floating Ideas', *Omega*, **13**, 331–347.
13 KELLY, G. (1963). '*A theory of personality: The Psychology of Personal Constructs*', New York: Norton.
14 WHORF, B. (1956) '*Language, Thought and Reality*' Massachusetts Institute of Technology Press, Cambridge.

A study of interactive computing at top management levels

C.J. Martin
Durham University Business School

Abstract

Computers are increasingly becoming part of the manager's personal environment. As computer use spreads throughout organizations there is a need to understand the ways in which managerial work is being affected. This paper describes research which examines aspects of the adoption and use of interactive computing by senior managers in large organizations. A behavioural model is proposed which represents the interplay of complex situational factors which seem to be involved in computer adoption processes at top levels, and some conclusions are drawn about the relationship of the computer to managerial work.

Introduction

It has become apparent in recent years that many organizations are making increasing use of computer technology at many levels, and in a widening range of applications. In particular, the development of managerial computing has seen increasing numbers of people directly involved in computing for their own personal use, and the trend seems to be towards more senior executives exploring the possibilities of direct computer use for themselves. This trend has a focal point in the technology of the Decision Support System (DSS) movement, which is aimed (at least in part) towards a form of manager/computer symbiosis such that the manager becomes more effective in harness with a computer system. This latter idea was expressed succinctly by Scott Morton in 1971:

> 'We may in fact be reaching the point where the manager in combination with the computer is going to be a significantly different and more effective manager than one without the computer ... it is far from clear whether or not the manager alone will be able to exercise effective control in the years to come.'
> (Scott Morton 1971)

The question arises as to whether, in the intervening years, such a combination has indeed been achieved at top management levels, and if so what have been the consequences for management roles. In particular, have top managers been obliged to change their jobs fundamentally as a consequence of the availability to them of computer systems? Is there a need for them to adapt their roles to accommodate special advantages offered by the new technology?

Prior research

In practice managers normally make use of computer-based information and facilities through intermediaries, a method which can be referred to as 'indirect' use. In the case of strategy-oriented Decision Support Systems for example, surveys by Grinyer and Wooller (1975), and Grinyer (1983) indicated that these systems are generally built and used by staff specialists or consultants rather than by the senior managers who might be expected to be the real consumers of the system outputs. However, the focus of attention in the present research is specifically on the top manager as a direct computer user, in the sense that it is he himself who operates the keyboard and other devices in order to interact directly with the information resources made available by the system.

Although there have of course been many studies of the impact of computing in organizations, there is little material currently available which reports directly on top management's personal computer use. There is however an underlying enthusiasm for the possibilities: for example, a report in Management Today (1984) quotes a survey which anticipates 'widespread use of workstations among senior executives' and several reports have appeared which state that computers are about to 'invade the executive suites' of one organization or another. According to Rockart and Treacy (1982) many top managers have begun to use computers themselves in order to carry out analyses for their own decisions, and these authors cite several examples.

What seems to be missing from these accounts is evidence (and particularly comparative empirical evidence) which would provide a reasonable understanding in behavioural terms of why some top managers apparently do use computers personally while the great majority, apparently, do not. It is of course possible that many managers at the very top of their organizations have not yet been directly exposed to interactive computing. However, there are a number of managers who have had such involvement and this study was designed to examine the circumstances and nature of their experience.

Although there is no single behavioural theory which is directly applicable to the topic under study, there is a substantial body of knowledge which examines factors relating to computer use generally and much of it can be found within the behavioural study of implementation problems. The difficulty inherent in implementing innovative management techniques in organizations has been recognised from very early on:

> 'In implementation we deal with the most difficult subject matter confronted by science: people and social groups.'
> (Churchman, Ackoff and Arnoff 1957)

Studies of computer implementation variables have examined a number of areas which may have a bearing on computer adoption and use. Kling and Gerson (1977) have identified a wider computing community which generates internal pressures on computer users; a view which contrasts sharply with the more conventional view that computer adoption is necessarily demand-led, or needs-led. Situational variables, such as user involvement in system design, have been discussed by a number of authors, but notably McKeen (1983) and Ginzberg (1981a) who identify user involvement and commitment as important in implementation success. In practice however, things may be rather different; a study by Alter (1978) identified patterns of user involvement which had overtones of compliance, and Lawless (1982) has described the key role of the middle-management advocate in facilitating and manipulating innovative organizational implementation mechanisms.

Attitudes and expectations have been researched quite extensively in the context of computer implementation. Lucas (1975, 1978) in particular has made a special study of attitudinal components, although the nature of attitudes is such that it can be seriously questioned as to whether there is any simple and direct relationship between attitudes and behaviour (Fishbein 1972). However, a study by Ginzberg (1981b) indicates the importance of pre-implementation expectations in relation to subsequent implementation success and this latter aspect was found to be significant in the present research.

Despite the great number of studies examining various aspects of the implementation problem there is still no well-developed and generally accepted implementation theory which accounts for all the possible variables. Neither is there any substantial work which addresses the special characteristics of senior managerial behaviour. In these circumstances it was felt appropriate to undertake an exploratory study which concentrated on identifying significant circumstances with a view to developing a new model which focussed on appropriate aspects of senior managerial behaviour specifically.

Scope and methods of the present study

The study set out to explore the nature and circumstances of senior managerial computer use in terms of management roles and activities, with particular reference to adoption processes. Semi-structured interviews were conducted with over 60 executives in UK organizations from public and private sectors. The study was limited to executive board members from private companies and the top tier of officers (or their equivalent) from public sector organizations. The survey included a small sample of owner-managers from small businesses (see Martin 1985a), and also of organizations where computing was not readily available to the intended respondent group; these latter groups provided useful comparative data but will not be discussed in detail here. Details of the groups surveyed are shown in Tables 1 and 2 in the appendix.

Research access to top managers is usually considered to be especially problematic; for example Pahl and Winkler (1974) describe how a high proportion of unsponsored approaches by researchers may be refused, and mention access rejection of about 85%, even with personal introductions. In order to overcome some of the practical and methodological difficulties a strategy was employed whereby once suitable organizational access had been achieved, strenuous attempts were made to see all the top managers in that organization. In practice, the research effort resulted in better than two-thirds of target respondents interviewed from five large organizations, which became the primary source of information used in this paper (see Table 1 in the appendix). A more detailed description of the access and interviewing methodology is contained in Martin (1985b). In addition to interviewing the senior managers themselves, information was gained from interviews with systems development and implementation personnel, from the study of internal reports and memos, and from investigating relevant computer systems and associated documentation.

The case study data enabled comparisons to be made between managers from different organizations, and perhaps more importantly, between managers from the same organization. In this way it was possible to compare data from respondents who were using a computer with data from others in the same organization who had access to the same facilities but were not using them. It was these latter comparisons which were particularly important in answering questions about the nature and circumstances of the managers' computer adoption and use.

The semi-structured interview questions to the senior managerial respondents related to four principle areas (see Table 3 in the appendix for details):

background and personal factors
previous computing exposure
system introduction processes
present computer use

A process model of direct computer adoption

It became apparent from the study that direct computer use is not necessarily a constant phenomenon in the sense that managers are simply either users or non-users. Instead, several managers in the survey had been involved in direct computer use at some point but had subsequently discontinued their use. This behaviour can be described in terms of a cycle of computer adoption and trial followed by discontinuance. Some managers in the sample had been through two or more such adoption/discontinuance cycles. Of course, not all managers in the survey had adopted personal computing at the outset, and on the other hand some who did so continued to utilise their systems to a greater or lesser extent. A process model which is proposed to describe these events is shown diagrammatically in Fig. 1.

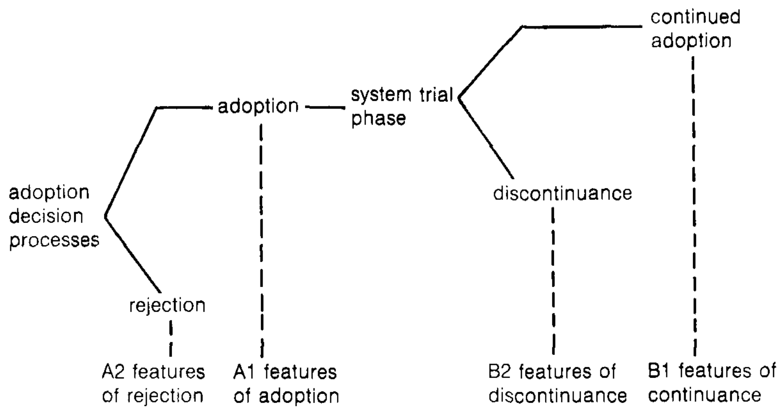


Fig. 1 A model of the managerial computer adoption process

A: Adoption/Rejection
B: Continuance/Discontinuance

The proposed model shows an initial decision process which results either in adoption of a computer system, or in rejection. If the system is adopted, then there is a trial phase followed either by continuing use, or discontinuance. All the managers in the research sample could be described in terms of their trajectories in these decision paths. The crucial feature of the model is that it postulates two different sets of variables which come into play at the two key points in the adoption cycle. Hitherto, it has been assumed that there are variables which generally influence computer use; here, it is specifically hypothesised that the set of variables which influences initial adoption is substantially different from the set of variables which influences subsequent continuance.

Returning to our earlier theme, the key questions about senior managerial computer use revolve around the circumstances under which some managers reject direct computing when others adopt it, and why some managers continue with their system use when others discontinue. Utilising comparative data derived from the research study, certain features of their circumstances can be associated with the managers' decisions at the two key points (A: Adoption/Rejection, B: Continuance/Discontinuance) in the adoption cycle.

A1 Features of initial adoption

1 High and middle-level advocacy

Managers were influenced by others within the organization at various levels and rarely was the decision to adopt an individual's lone choice. Although it might be supposed that top level managers have complete discretion over these choices, in practice they usually described the ways in which they were influenced by others, and in one or two instances this amounted to compliance rather than whole-hearted enthusiasm.

2 Leadership role enactment

Nearly all the managers referred to their leadership roles when describing their computer adoption. In some instances this occurred in the context of a requirement to lead their direct subordinates with respect to computer use; in others because they were in charge of a division with responsibility for the computing resource, and hence there was a need to show appropriate usage leadership to other divisions on behalf of the manager's subordinates. Even when there were no direct functional responsibilities of this kind, senior managers described the requirement for them to reflect positive values with regards computing to others both within and without the organization.

3 Expectations of particular rewards and benefits

Managers had certain expectations with regard to the computer systems which they would receive, in terms of personal informational benefits. These expectations varied, depending partly on the manager's prior experience of direct computing; where the manager had no previous experience the expectations were often unrealistically favourable, leading to subsequent expression of substantial disenchantment and disappointment following system trial.

A2 Features of non-adoption

In most respects the features of non-adoption reflected a corollary of the features of adoption. In a few instances, managers had specifically rejected direct computing for themselves even though their board colleagues had adopted. Either the internal pressures to adopt had been less acute in their particular circumstances, or they had specifically resisted the pressures. With regard to their leadership requirements, either they perceived there to be no requirement for them to display pro-computing values, or, as in one instance, the manager perceived that computer use would reflect negatively on his position.

B1 Features of continuance

1 Polarised system usage patterns

A number of managers in the study were presently utilising their computer systems to one degree or another. It was noted that a significant characteristic of their use was the tendency to stick with one particular system or facility, even where a wide range of facilities was available. This was taken to be a response to the need to minimise the personal time and effort costs which were associated with learning how to use new systems.

2 Successful interweaving of system use into the manager's present role set

A significant feature of the continuing use was the direct utilisation of a system within the manager's present work activities. For example, personnel directors used their terminals to access data on individuals from computerised databases; this activity would have been carried out before computerisation using record cards or other means and the automation represented a straightforward replacement of one access mechanism for another. Other top managers who used their systems regularly had all found some way in which their computer use related to current activities. None of the respondents made any significant use of generalised decision support systems, even where these systems were very extensive and sophisticated. In general there was little evidence that any manager's work roles had changed significantly as a result of his computer use. All respondents were asked about the way in which direct computer use had altered their personal work activities, and most of them denied that there had been any significant change.

3 Lack of effective informational advantage

What was particularly noticeable was that systems use did not appear to confer any particular informational advantage on the user (and some systems were compared unfavourably to alternative access methods). In particular, many of the top managers in the study could command significant human informational resources in terms of staff advisers, financial analysts, and so forth. The direct computer facilities were considered to be poor alternatives to such resources except under special circumstances. In many instances, continuing computer use seemed to be related to the achievement of other goals, particularly as regards the influence of others or the expression of values with respect to the man and his organization.

B2 Features of discontinuance

A number of managers had discontinued their computer use after a trial of the facilities available to them. Analysis of the circumstances of this discontinuance was particularly revealing.

1 Difficulties at the man/machine interface

All the discontinuing managers described difficulties in terms of time and effort costs associated with learning to use computer systems. It must be emphasised that many of the respondents had made very determined efforts to utilise their systems, and had acquired substantial knowledge about the system before ceasing to use it. For many respondents, the value of the information to be had from the system seemed to represent a poor reward for the considerable efforts required to access it.

2 A mismatch between system intentions and managers' real role requirements

It has been found most useful to consider the computer system as being aimed at addressing various aspects of the manager's informational and decisional roles; it appeared that the systems failed to do this successfully because the systems were incorrectly oriented as regards the particular nature of senior managerial work in respect of these roles. In particular, where the information was aimed, for example, at monitoring or resource allocation roles it often reflected issues which would be of more interest at lower management levels. Apart from the orientation failure, it seemed that this computer information just could not compete with the information sources which the managers had developed from their existing personal networks.

This is a key point: it is a characteristic of senior managers that they acquire and develop superior information sources through skilful manipulation of interpersonal networks (Kotter 1982). Attempts to create and implement systems which were to compete directly with these sources (even formally based ones) seemed to have been ineffectual. In a similar way, the managers' liaison and dissemination roles were usually not enhanced by computerised messaging systems partly because the systems did not include important contacts who were part of the manager's interpersonal networks, and partly because they lacked the immediacy of face to face contacts or the efficiency (for the managers) of traditional methods.

Discussion

It was possible to analyse the computer systems which had been made available in terms of the managerial roles which the systems were intended to address. A useful representation of the different roles enacted by managers has been developed by Mintzberg (1973), who has identified 10 different managerial roles: figurehead, leader, liaison, monitor, disseminator, spokesman, entrepreneur, disturbance handler, resource allocator and negotiator. It is of course necessary to point out that Mintzberg's theory, as it relates to these ten specific roles, is not without its critics; in particular Luthans, Rosencrantz and Hennessey (1985) cite studies which do not show direct empirical support for categories. However, it can be argued that what is important here is not whether certain specific senior managerial roles are universally prominent or not, but rather that it is recognised that a number of very different roles exist in managerial work, many of which are not directly concerned with 'decision making' or 'information processing' as these are commonly understood.

Referring to the role structure of Mintzberg, it can be shown that the computer systems in the study were intended to support the manager's decisional and monitoring roles primarily, although in some cases messaging systems were provided which were intended to support liaison and disseminator roles. The other interpersonal roles of figurehead and leader were not intentionally addressed by any of the systems, even though for senior managers such roles might be expected to be especially significant. In practice, the managers considered the consequences of adoption in terms of their own leadership role most carefully, and it seemed that for most of the respondents these considerations were more important to them than any other in their adoption decision.

In the second phase of the adoption cycle, the manager's decision to continue his computer use (or to discontinue it) seemed to be based on a balance between the effort and rewards occasioned by the initial and ongoing effort costs on the one hand, and the informational and behavioural benefits on the other. For many respondents in the study the former far outweighed the latter. The important point is that simply improving the man/machine interface aspects, for example, will not necessarily help matters; if the system information remains incorrectly oriented, or represents bad value for other reasons, then the manager will perceive that his efforts are not sufficiently rewarded and he will discontinue, however 'user friendly' the interface may be. This is illustrated by the fact that many of the managers in the sample had made strenuous (and largely successful) efforts to overcome considerable interface and learning difficulties, but had then discontinued when they found out for themselves the precise nature of the system's real information value in terms of their real roles.

Summary and conclusions

The two-phase behavioural model of the computer adoption process provides a framework for considering key aspects of the circumstances of computer use by top managers. It appears that the nature of managers' role requirements is a key factor in understanding these circumstances, both at the initial adoption phase and also in identifying reasons for subsequent usage or discontinuance patterns.

There was little evidence from the study to support the idea that the roles of top managers had been profoundly altered owing to the availability of direct computing facilities. On the contrary, where a manager made significant use of direct computing, it appeared to be the case that he had been able to interweave the computer activity into the fabric of his present role set and work activities without substantial dislocation of his traditional functions. Neither was there evidence of distinct informational or decisional advantage provided by the systems, in the sense intended by the system designers.

Where managers had discontinued their computer use, this was associated not only with significant difficulties (and hence personal effort costs) at the man-machine interface, but also with a failure of the systems to provide adequate rewards which would offset those costs. On the one hand the computer systems addressed limited aspects of the manager's role set, and on the other there appeared to be errors of orientation with respect to the top manager's special informational needs and activities.

Despite the substantial mismatch between the presently available systems and the realities of senior managerial roles and behaviours, it seems inappropriate to advocate that the managers should change their behaviours in order to accommodate the technology. It remains therefore a substantial challenge for the management computing and decision support community to devise future computer systems which can successfully address key aspects of top managerial work and hence offer significant advantage to direct users at this level.

Acknowledgement

The author gratefully acknowledges the financial support of ICL Knowledge Engineering Centre, Manchester.

References

ALTER, S.L.: Development Patterns for Decision Support Systems, MIS Quarterly, Sept. 1978.
CHURCHMAN, C.W., ACKOFF, R.L., ARNOFF, E.L.: Introduction to Operations Research, John Wiley, New York, 1957.
GINZBERG, M.J.: Key Recurrent Issues in the MIS Implementation Process, MIS Quarterly, June, 1981a.
GINZBERG, M.J.: Early Diagnosis of MIS Implementation Failure: Promising Results and Unanswered Questions, Management Science, Vol. 27, 4, 1981b.
GRINYER, P.H.: Financial Modelling for Planning in the UK, Long Range Planning, Vol. 16, 5, 1983.
GRINYER, P.H. and WOOLLER, J.: Computer Models for Corporate Planning, Long Range Planning, Feb., 1975.
KLING, R. and GERSON, E.M.: The Social Dynamics of Technical Innovation in the Computing World, Symbolic Interaction, Vol. 1, 1, 1977.
KLING, R. and GERSON, E.M.: Patterns of Segmentation and Intersection in the Computing World, Symbolic Interaction, Vol. 1, 2, Spring, 1978.
KOTTER, J.P.: The General Managers, The Free Press, New York, 1982.
LAWLESS, M.W., et al: Enhancing the Chances of Successful OR/MS Implementation: The Role of the Advocate, Omega, Vol. 10, 2, 1982.
LUCAS, H.C.: Why Information Systems Fail, Columbia Free Press 1975.
LUCAS, H.C.: Empirical Evidence for a Descriptive Model of Implementation, MIS Quarterly, Vol. 2, 2, 1978.
LUTHANS, F., ROSENCRANTZ, S.A., HENNESSEY, H.W.: What Do Successful Managers Really Do? An Observational Study of Managerial Activities, Journal of Applied Behavioural Science, Vol. 21, 3, 1985.
Management Today, Data Management Supplement, Sept., 1984.
MARTIN, C.J.: Information Technology in the Small Firm: The Role of the Chief Executive, a paper presented at the Eighth National Small Firms Policy and Research Conference, N. Ireland, Nov., 1985a.
MARTIN, C.J.: Accessing and Interviewing Senior Managers, Graduate Management Research, Vol. 3, 4, 1985b.
McKEEN, J.D.: Successful Development Strategies for Business Application Systems, MIS Quarterly, Sept., 1983.
MINTZBERG, H.: The Nature of Managerial Work, Harper & Row, 1973.
PAHL, R.E. and WINKLER, J.T.: The Economic Elite: Theory and Practice, in Stanworth and Giddens (Eds), Elites and Power in British Society, Cambridge University Press, 1974.
ROCKART, J.F. and TREACY, M.E.: The CEO Goes On-Line, Harvard Business Review, Jan./Feb., 1982.
SCOTT MORTON, M.S.: Management Decision Systems, Grad. School of Bus Admin. Harvard Univ., Boston, USA, 1971.

Appendix

Table 1 Organizations and Respondents

		Respondents		%
		Interviewed	Target	
1	ABC Multinational	6	7	86
2	BCD Multinational	10	15	67
3	PHA Public Corporation	5	6	83
4	CCC County Council	4	6	75
5	NHL Institute	7	10	70
6	Small firms group	16	20	80
7	District Council officers	11	12	92
8	Others/miscellaneous*	8		
		67		

*This category represents single respondents from organizations where further access was not possible for one reason or another.

Table 2 Frequency distribution of sample respondents

	Group 1 Private sector	Group 2 Public sector	Group 3 Small firms	Group 4 Misc. group	Totals
Pre-adoption/small firm groups		15	10	1	26
Adopted: pre-trial		2		1	3
Rejected: pre-trial	1	4		1	6
Discontinued users	9	1	4		14
Slight users	2	2		1	5
Moderate users	2	2		1	5
Heavy users	2	1	2	3	8
Totals:	16	27	16	8	67

Table 3 Topics Addressed in the Interviews

1 Personal factors
   - Position in organization
   - Responsibilities
   - Career history, age, education

2 General computer exposure
   - Formal training
   - Past experience
   - Home computer ownership and use

3 Computer Facilities in the organization
   - General facilities
   - Facilities available to the subject

4 Category of user
   - Non user, discontinued, slight, medium, heavy

5 Factors in adoption

6 System introduction processes
   - Introduction process
   - Subject's relation to design/choice processes
   - Training facilities and use
   - Manuals and written materials
   - Time spent on learning

7 Computer System Usage
   - Systems use (amount per day)
   - Nature of use
   - Most used/preferred systems/what is of most use
   - Why used personally
   - Perceived difficulties/disadvantages

8 System use in relation to job and job roles
   - Job facilitation
   - General relevance
   - Role changes/job changes related to adoption
   - Relation to strategic decision – making

9 Changes in perceptions of value of systems
   - Views expressed after experience
   - Views expressed before experience (4 subjects only)

10 Factors in continuance and discontinuance

A management support environment

Noel C. Austin
ICL Management Support Business Centre, Reading, Berkshire

Abstract

In June 1986 the Management Support Business Centre of ICL Applied Systems launched the 'pod', a meeting environment for managers and professionals. The launch was the culmination of a programme of research and development which started when the Business Centre was established in May 1983 and became visible with the opening for use of the first, prototype pod in Kings House in March 1985. This paper:

- outlines the research which was carried out and describes some of the key results of this research
- suggests a diagrammatic model of the use of information in organisations which explains the potential role of the pod
- provides a physical and functional description of the pod
- discusses some of the cultural and organisational issues which surround any decision by an organisation to install a pod
- discusses the question of staffing a pod
- outlines the support services which ICL provides
- describes the benefits which have already been experienced by users of the existing pods

1 Research programme

Two main streams of research contributed to the programme which led to the creation of the pod; one of these was devoted to establishing the areas to which senior managers were beginning to address their attention and the other sought to establish the characteristics of products and services which might support those senior managers in the business of managing their enterprises.

The companies whose successive mergers led to the existence and growth of ICL included organisations which had been present at the birth of the commercial use of computers. The company had, and still has, a pride in the technical excellence of its products, and unsurprisingly the company's management was technically very strong. Together with a preoccupation with the inevitable structural issues resulting from years of acquisitions and a commitment to a geographically widespread market place, this had left little time for management development. The Management Support Business

Centre (MSBC) came into being in May 1983, a couple of years after the appointment of Robb Wilmot as Managing Director. At that time the company was experiencing a management renaissance, which has been graphically described in various management journals, culminating in a recent series of three articles in the 'Financial Times'; it was changing from management based on the application of pragmatic, experiential knowledge to tactical problems to management which had begun to exploit the enormous upsurge in strategic management thinking in University Business Schools in North America, in the UK and in continental Europe. Management was no longer seen as an exact science which was frustrated in its application by the intransigence of the real world, but as the art of exploiting the organisation's strengths to take advantage of the opportunities presented by an ever-changing and contradictory external environment. It was within this context, catalysed by the recently established relationship with the Alfred P. Sloan School of Management at the Massachusetts Institute of Technology, that members of the Business Centre began an on-going research programme into the concerns of managers and the practice of management.

Over a period of some months, contact was established with Schools at several British Universities, notably the School of Management at the University of Bath, the Decision Analysis Unit at the London School of Economics and the Department of Systems at the University of Lancaster, all of which were working on various aspects of defining and supporting management decision areas which they saw as necessitating an understanding, inter alia, of group behaviour, unstructured or semi-structured problems and hard and soft information. Each of these early relationships has developed into collaborative ventures, with a joint ICL-LSE venture into Decision Conferencing and the further development of MAUD, members of the University of Bath becoming involved in ICL's strategic thinking and a member of the Business Centre undertaking part-time post-graduate research under the supervision of a member of the University of Lancaster. The relationship with the Sloan School has resulted in MSBC becoming closely involved with the 'Management in the Nineties' programme and the involvement of members of the School in MSBC's recently established Senior Management Programme.

However, there was one area in which the Business Centre was unaware of any UK-based research taking place – the subject of the trends which were likely to affect the future conduct of business. In the United States John Naisbitt had published 'Megatrends', a book based on a survey of trends which used as its material the frequency with which subject matter was referred to in the US press. This research was thought to be of doubtful value because several of the identified trends appeared to be at least partly based in the American culture and because it was felt that since the Business Centre's concern was with management views, the perceptions of the American press might be a less than reliable indicator. The decision was therefore taken to gather views direct from senior managers in British industry, commerce, local government and national government together with a small number of bodies whose interests included aspects of future trends.

The survey, colloquially and, as it subsequently turned out, misleadingly referred to as 'The Business Trends Survey', took the form of a series of about fifty interviews with senior managers in which they were asked their views about the trends which were likely to impact on the way businesses were managed in the next five years, and then were asked to prioritise the trends they had identified, No attempt was made to constrain the areas in which trends were defined. Each interview was written up in detail and a combination of spread-sheet based and cognitive mapping techniques was used to identify and group the main trends. The detailed results of this survey have been documented and published within ICL and cover a very wide range of political, economic, social, technological and cultural trends but a small subset of them referred to senior managers' concerns about Information Technology.

Some of the most frequently mentioned and highest priority trends were:

- the fact that information technology was in the process of negating the assertion that 'information is power' and that in the future, power will be vested in the people who have the expertise and the technology to exploit information
- the failure of information technology to supply senior managers with the information they needed to manage, particularly when group activities were involved
- the explosion in the number and variety of sources of external data and the difficulty of knowing which were relevant
- the shortening of strategic timescales to such an extent as to force managers to think anew about what was strategic
- the exponential growth of complexity in the problems top managers were being confronted with.

The results of these research programmes suggested that ICL might be in a position to secure a strategic market advantage if it were to begin to pull together the emerging soft decision making technologies, including group behavioural theory, and use them to address some of the concerns which top managers expected to occupy them during the ensuing few years.

The other research results which made a major contribution to MSBC's thinking were those which related to group behaviour. Much of this material is now in the public domain but has not been much exploited by designers of management environments. The key results which were particularly relevant to the design of a senior management meeting room included:

- the optimum creative contribution occurs with seven attendees present in a meeting
- a round table creates a perceived equality between the attendees and this tends to increase their willingness to participate

- internal and external, visual and audible distractions, such as passers-by and obtrusive technology, have an adverse effect on concentration
- restful decor and lighting encourage creativity and concentration
- music played quietly in the background encourages creativity and tends to blank out extraneous external noises
- graphical presentation of numerical information speeds absorbtion and increases understanding
- IT should be flexible and easy to use.

At the completion of this research programme it became clear that the way forward should now be the construction of a prototype. It was proposed to construct this prototype in Kings House, Reading, the home of MSBC, and so the Business Centre approached Lucas Furniture Limited, the company which has supplied the furniture with which the building had been equipped. Within a short time it became apparent that collaboration between the two organisations was potentially very synergistic, as the senior management of Lucas had a strategic commitment to involvement with the information technology industry. The design and construction of the prototype pod was carried out jointly with Lucas Interiors Limited, another company in the Lucas group, and with the subsequent involvement of Audio Visual Materials Limited, an audio visual systems contractor. The prototype pod was designed, constructed and commissioned in an elapsed time of eleven weeks, a clear demonstration of the commitment of the parties to the project.

As the pod was ICL's first encounter with the problems of designing a high technology meeting environment, it had been expected that the lessons learned in designing, building and using the prototype pod would reveal that is was deficient in a number of respects and would lead to its demolition and replacement by a facility significantly different from the prototype. This did not prove to be the case and, with the exception of a much more sophisticated control system the pod today is very similar to the prototype.

2 Information in organisations – a diagrammatic model

A development of the familiar organisational triangle diagram can be used to relate the quantity and quality of information used by a manager to his level in the organisation within which he works. It is a simple model, developed for use in a management consulting environment as a basis for discussing with senior managers their information needs and the kinds of solutions which may satisfy them; it has proved to be widely accepted and easily understood and provides a means of setting the pod in context in the IT solution space.

Fig. 1 shows the conventional organisational triangle; it is a model of a strictly hierarchical organisation, and is therefore in most cases a gross generalisation. It indicates the relationships between the levels of management in a typical organisation and also suggests the numbers of managers present at each level. However, the addition of the two dotted lines in Fig. 2 can be used to show that senior managers have overlapping interests in the activities of the organisation.

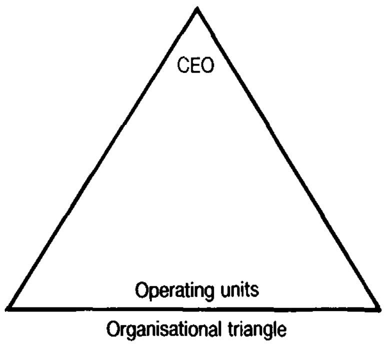


Fig. 1 The organisational triangle

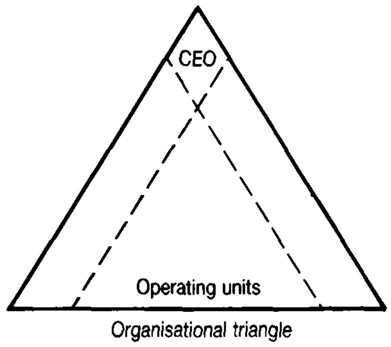


Fig. 2 The organisational triangle showing the overlapping interests of senior manager

The model which forms the basis of the remainder of this part of the paper is created by superimposing on the organisational triangle a further, inverted triangle, the 'information triangle', as shown in Fig. 3.

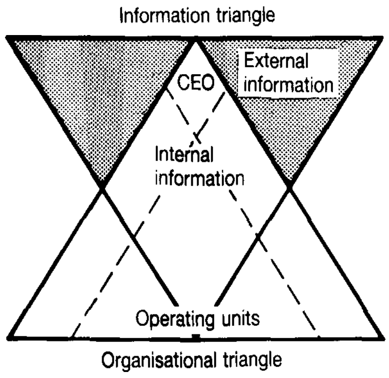


Fig. 3 The information triangle

In this diagram, the inverted triangle represents the body of information available to managers within an organisation. Those areas within both the organisational triangle and the information triangle represent internal information, information about the organisation and its operational activities, whereas those areas within the information triangle but outside the organisational triangle represent external information, information about the organisation's external environment.

At the bottom of the organisational triangle, supervisory and junior line managers manage using a very restricted subset of the organisation's internal information, often consisting of data at account or transaction level, and make only very limited use of external information, if at all. Higher up the organisation, middle managers are usually interested in the gross performance of component operating units, taking a statistical rather than a deterministic approach; they become interested in individual accounts or transactions only when these are large enough to cause perturbations in the performance of an operating unit. At the top of the organisation, the Chief Executive Officer is interested only in a very highly summarised view of the organisation's internal information, but with a heavy emphasis on external information.

This external information is significantly different in source and character from the internal information:

- it is seldom contained in the organisation's own computer information base
- it comes from a wide variety of external sources, such as commercially available databases, books, newspapers, magazines, market research organisations, radio and audio recordings, television and video recordings, discussions and overheard snatches of others' conversations, to name but some
- it is therefore carried on a wide variety of media
- it may take the form of data, text, image or voice, or any combination of these
- its reliability is variable
- its relevance is debatable
- it may sometimes be security classified.

The information available to a decision maker plays a large part in determining the way in which he makes the decision, as shown in Fig. 4. Supervisors and junior line managers generally make decisions about the detailed disposition of operational resources such as men and machines. The decision making processes are largely deterministic and the systems available to support them are becoming increasingly sophisticated; in many cases human intervention in this decision making process is continued only for historical and labour related reasons. Middle management decisions often require the decision maker to make judgements, but these are based on a statistical view of the organisation and on his knowledge of its capabilities;

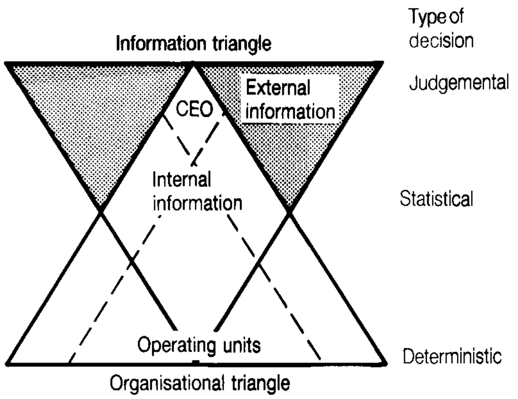


Fig. 4 Decision making approaches at different levels of management

when external information is used it is largely quantitative rather than qualitative. A study of the above list of characteristics of the external information used by top managers reveals that these have a fundamental effect on decision making. Even before setting out to make a decision, the top manager will commonly have to make judgements about:

- what the decision is actually about
- of the information he has, what is relevant and what is not
- whether he has sufficient relevant information, and if not, whether the necessary additional information is accessible to him
- how reliable is the available, relevant information
- what does this information mean when applied to the problem area
- what assumptions should he make about those aspects of the decision for which he does not have adequate information.

Only then can he reach a decision, which he may have to make on the basis of incomplete information of dubious reliability; as can be seen, such decisions rely heavily on the judgmental abilities of the decision maker. Yet such decisions are often strategic, even to the extent of determining the organisation's chances of survival.

The problem is further complicated by the fact that, in most medium to large commercial, industrial and governmental organisations, decisions are taken as result of peer group consensus rather than by individuals. It is often the case that, although an individual may have the formal authority to make a decision, he does not have the breadth and depth of knowledge to enable him to do so; furthermore, without the detailed involvement of those managers who have to implement the decision it is unlikely that the necessary commitment will be forthcoming.

It is an observable phenomenon that a group of managers each asked to make a decision about the same strategically significant problem will reach a range of different conclusions. Those conclusions are based on the individual's personal judgments, which reflect not only the information which he holds in common with his peers but also his own education, training, experience and preferences. Putting the managers together in a meeting will not, in general, ameliorate the situation, since even at the highest level managers fail to communicate because they use the same words to mean different things, and different words to mean the same things, and do not hold the same information in common. Furthermore, personal chemistry and the participants' views about the reliability and relevance of the opinions expressed by their colleagues obtrude into any group decision making process.

Now, let us examine, with reference to Fig. 5, the support which current IT-based systems are capable of giving to managers at each level for the types of decisions they have to make. Supervisors and junior line managers were among the first people to benefit from the introduction of data processing. Early commercial systems, in addition to taking over clerical tasks and enabling a substantial reduction in the clerical workforce, provided managers with early management information; for instance, payroll systems were able to provide labour costs by department and sales ledger systems indebtedness by customer with exception reports on customers who had exceeded their credit limits or were slow payers. This pattern has been continued to the present day so that few large-scale clerical tasks remain to be automated. If present-day business operational systems do not meet the needs of their users, the shortfall is more likely to be due to a lack of resource or management commitment than to the inadequacy of information or decision making technologies. The developments of the past five years, notably decision support systems and information centres, now provide many middle managers with support. Their needs for tools to help them identify and monitor trends, plan projects and make operational forecasts are now satisfied by statistical, modelling, planning, enquiry and knowledge based

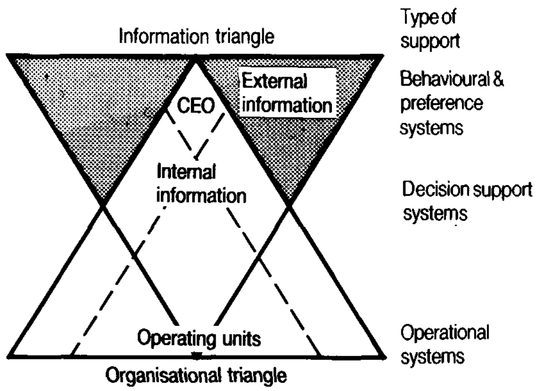


Fig. 5 Types of IS support for decision makers at different levels of management

systems, with the recent rapid increase in the number and variety of external information databases providing them with the quantitative external information they need for modelling and forecasting. The shortfall between desire and reality is due to the inaccessibility of corporate information combined with under-investment, conservatism, lack of training and lack of time.

But what about systems to support the kind of senior management decision making described above? The quick answer is that very few such systems yet exist, and those that do ignore one or more of the key characteristics of such decisions. A study of the characteristics described reveals that a system to support top managers must combine:

- access to information from a variety of sources, both internal and external
- and on a variety of media
- the means to present this information in a variety of ways to meet the wishes of the decision makers
- and to use it for modelling and to make enquiries about it
- so that all the decision makers may share the same information and insights
- in an environment which supports the kind of behaviour most likely to lead to a 'good' decision
- and is private, or possibly even secure.

It was in response to an understanding of this need that the concept of the pod originally arose. We shall see later in this paper that the pod satisfies all of the above criteria; although there is still substantial scope for improvement, the pod represents the state of the art in management support.

3 The pod – a physical and functional description

The pod consists of a combination of four types of elements:

- a physical environment
- information technology systems
- audio-visual display systems
- a control system.

The detailed specification and organisational and physical location of the pod as delivered to a particular customer reflect the needs of the organisation. As explained below, these are a function not only of the meetings profile but also of the corporate culture.

Physical environment

The pod is an octagonal meeting room within a square room with sides between 6 m and 7 m and height 2.5 m. It is of pre-fabricated, modular construction and requires a single 35 amp power supply; it may require air conditioning to be installed if the building in which it is situated does not

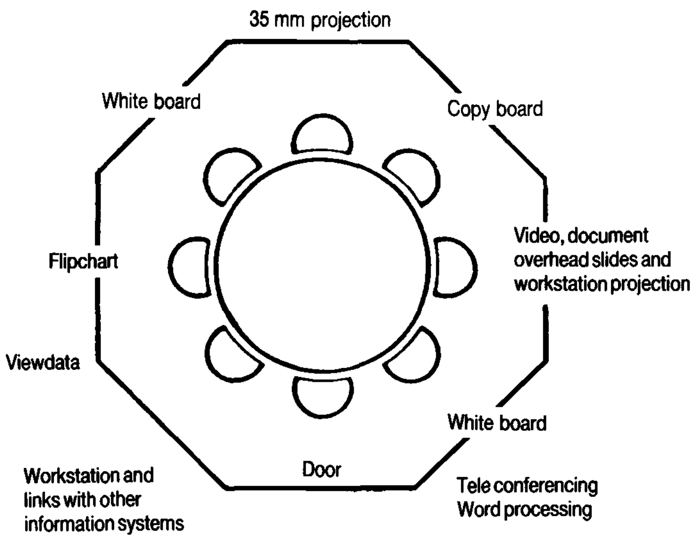

35 mm projection
White board
Copy board
Flipchart
Video, document overhead slides and workstation projection
Viewdata
White board
Workstation and links with other information systems
Door
Tele conferencing Word processing

Fig. 6 Schematic plan of a typical pod

already have suitable provision. As shown in Fig. 6, six faces of the room bear a writing surface or a projection screen and each is lighted independently; there is also general room lighting and all lights may be dimmed.

The pod is equipped with a circular table with eight chairs, and an Information Engineer's workstation occupies the space in one corner of the square; the remaining corners are occupied by storage and audio-visual components. The pod's decor is in restful colours, typically green and beige, different from but sympathetic to the decor of the surrounding environment. The lines of sight permit anyone sitting at the table to see what is written or projected on each face and the diameter of the table encourages one-to-one communication between participants.

Information technology systems

The IT systems within a pod must enable its users to gain access to and manipulate information held on IT systems both within and outside the pod and must support the forms of presentation selected by the users. Typically, a pod contains:

- a Personal Computer (e.g. and ICL PC or PC Quattro) with a graphics head
- a word processor (e.g. an ICL 8801)
- a One Per Desk terminal (OPD).

together with the hardware and software systems to permit access to externally stored information and data. In addition to ICL PCs, PC

Quattros, 8801s, DRSs and OPDs, the existing pods have at various times made use of ITT Xtras and IBM PCs, and have supported access to a variety of external information sources operating under both VME and various IBM operating systems. Application software used has included a wide range of PC-based systems such as WordStar, SuperCalc II and III, Illustrator and EGO, MAUD, COPE and a range of Decision Conferencing software tools.

Audio-visual systems

The pod's audio-visual systems support all the major forms of recording, playback and projection of audio and visual materials. They include:

- a television camera with zoom lens focussed on an area of the table with overhead lighting and in which is mounted a light box so that documents, overhead projector slides and even solid objects may be displayed and zoomed in on if desired
- a video projector which projects the documents and overhead projector slides and the contents of the PC, OPD, 8801 and DRS screens described in earlier paragraphs, together with video recordings and the usual television channels
- a video recording and playback system which can record all materials projected by the video projector
- an audio recording and playback system
- a 35 mm projection system
- a Muirhead 'Copy Board' which is a scrollable full-size whiteboard with five surfaces; facsimile technology is used to provide up to 99 copies of four of the five surfaces on demand
- three low-reflectivity white-boards
- a detachable wall-mounted flip-chart board.

The television camera, video projector and 35 mm projector are situated in a flat, fibre-glass or spun aluminium dome suspended from the ceiling above the central table. This removes the equipment from sight and, by eliminating the need for back projection, enables the pod to occupy a much smaller space than would otherwise be the case.

Control systems

In the prototype pod, all systems were controlled from a control board situated in the Information Engineer's work station but this was rapidly replaced by the hand held infra-red controller, illustrated in Fig. 7 and now supplied as standard equipment. It enables anyone sitting at the central table to control the lighting and audio visual facilities, the infra-red sensor being situated in the dome above the central table; the PCs, OPD, 8801 and DRS are controlled as usual using the appropriate keyboard. The control system permits:

- individual lights or all lights to be turned on or off or dimmed up or down or the selection of a small number of pre-set lighting levels

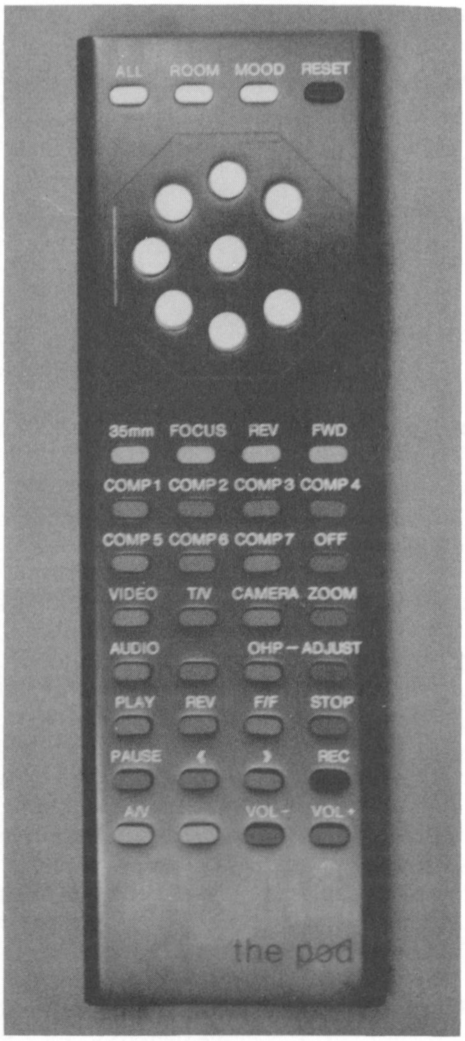


Fig. 7 A typical hand controller

- remote control of the 35 mm projection system
- any one of seven computer displays to be projected
- remote control of the video system
- remote control of the audio system
- the television camera to be turned on and off and zoomed in on documents or overhead slides.

There is also a reset button which returns the pod to its initial state if the user becomes confused about the state of the various facilities.

4 The pod and corporate culture

The subject of corporate culture is a complex one; both theory and early experience with prospective pod customers have revealed that it requires sensitive investigation prior to any intended pod implementation. The culture of a target organisation determines:

- whether it is likely to be able to benefit from the implementation of a pod
- to what use it is likely to be put
- its design
- its preferred organisational location
- its preferred physical location
- the type and quantity of resources needed to support it.

A decision by an organisation to acquire a pod, or any other senior management facility visible to employees in the organisation, constitutes a public commitment to a particular style of management. In the case of the pod, commitments to participative management and to the exploitation of information technology are necessary but not sufficient. There are many situations, such as a strictly hierarchical organisation or one in which information technology is a grudge purchase, into which it would be pointless to put a pod. There are, however, a number of organisations in which management practice already reflects the kind of thinking discussed earlier in this paper and these merit further exploration. One such organisation, surveyed prior to the formal launch of the pod as a product, was the UK operation of a large American company with a published statement of corporate culture which included commitments to participative and open management; on further investigation it emerged that 'open management' actually meant visible management, to the extent that even the Board Room had glass walls and, in the US head office, no doors. In such a context it was clear that the pod, with its enclosed environment, would run directly counter to the corporate culture. The lesson of this experience has been built into pre-implementation study.

The ways in which meetings are organised and take place in an organisation (the meetings profile) prior to the implementation of a pod have a strong influence on the pattern after implementation, and a study of the meetings profile is one of the tasks undertaken during the pre-implementation study. This normally takes the form of a detailed survey of all meetings taking place in one or two key internal conference rooms over a period of about a month together with any regular meetings taking place in the same rooms outside that period. The survey identifies, for each meeting:

- its purpose and structure
- its date, time, duration and frequency (brief, *ad hoc* meetings are commonplace in many organisations)
- who organised it (e.g. name, job function), for whom, and when
- who attended (e.g. name, job function)

- what information was used
- what was the source of that information (e.g. corporate mainframe computer systems, external databases, PC models, newspapers, magazines, TV programmes)
- what medium was used to present it (e.g. overhead slides, documents, 35 mm slides, video film)
- what form of presentation was ueed (e.g. tables, graphs, histograms, pie charts, text pictures, models)

This data, combined with the views and intentions of the pod's prospective users, supplies the information on which the functional design of the pod is based.

Potential users of the pod are drawn from a wide variety of industrial, commercial, financial, governmental and academic institutions and the case which each body might use to justify the acquisition of a pod is heavily dependent on the way in which it views its internal organisation and how it positions itself within its environment. The following cases, some drawn from experience and some conjectural, illustrate this argument and also show how it may point to the proper organisational and physical location.

- An organisation in, say, a sector of the nuclear power industry, which is the subject of the intensive interest of environmental pressure groups, has a continuing need to be able to present its case to those pressure groups and to the press, since its very survival might depend on its ability to convince them. In such a situation the pod might be used for carefully planned and well rehearsed presentations to and meetings with influential journalists and politicians and other public figures. In such a case the pod might be situated in a prestige position in the London Head Office and "owned" by the public relations function.
- Business Schools put significant effort into teaching group psychology and team working, and psychology teachers might use a pod for group behavioural teaching and experiential learning, so it would be sited in a teaching block
- Advertising agencies and public relations consultancies might use a pod for presenting proposals to their clients
- The management of a large manufacturing plant might use a pod for performance reviews and for monitoring large scale implementation and maintenance projects, so it would be sited in the factory management suite
- A management consultancy might use a pod for assignment reviews, for pooling the findings of assignment team members and for briefing clients, and would site it in a local office.

The question of support to the users of a pod is discussed later in this paper.

5 Staffing a pod

The pod has been designed to be easy to use, and experience has shown that all but the most technology-shy managers demonstrate an easy familiarity with the facilities of the pod once they have spent half an hour 'playing with' the hand controller. However, exploiting the pod to the full requires not only familiarity with its facilities but also an understanding of the processes which go on there and an understanding of the information used by managers in discharging their various group activities. It is for this reason that under most circumstances the Business Centre recommends the appointment of an Information Engineer.

The title 'Information Engineer' has, like many other job titles in the IT industry, been debased by careless usage. The person to whom we refer here is someone who, when supporting a group of senior managers in the pod, can understand the topics under discussion well enough to be able to anticipate the information needs and provide the information needed just as the participants are about to ask for it. This implies that he or she:

- is business trained, probably a business school or economics graduate
- is information technology literate
- is an expert on internal and external information bases and libraries
- is knowledgeable about the organisation and its practices and procedures
- enjoys the confidence of the senior managers with whom he is working.

Such a person is likely to be already employed as Personal Assistant to one of the other senior managers involved in the meeting.

6 Supporting and maintaining a pod

Four types of support are provided for the pod:

- pre-implementation planning
- implementation project management
- customer training
- post-implementation support and maintenance.

ICL acts as prime contractor for the supply, installation, commissioning and post-implementation support, with Lucas Interiors Limited and Audio Visual Materials Limited acting as sub-contractors in each stage. The pre-implementation planning process, which includes the survey and contract preparation, has been described above.

The implementation project management service is straightforward project management, covering the life of the project from immediately after contract signature until the completion of customer training. It includes such tasks as liason with the customer's building services department, monitoring the subcontractors against the plan and resolving any problems or conflicts which occur during the project.

Customer training takes three forms:

- general familiarisation

– user training
– technical training.

The familiarisation course is a two hour introduction to the pod designed for everyone who is likely to be involved with it, in whatever capacity. It includes a simple introduction to the reasoning behind the pod, a demonstration of the information technology facilities and control system and a question and answer session.

User training, which is designed for people likely to plan and run sessions in the pod, including the Information Engineer, if appointed, develops the material included in the familiarisation course, gives hands on experience in using the control system and gives advice and guidance on how to structure a meeting and exploit the facilities of the pod, related to the type and objectives of the meeting being organised.

The technical training course is designed for the one or two people who will be responsible for the day to day operation of the pod, who will be called upon to perform any technical tasks such as making adjustments to the audio visual systems and carrying out file maintenance on the pod's PC and OPD and who will be the first line of technical liason with ICL.

ICL acts as the prime contractor in the maintenance contract but makes use of the services of Audio Visual Materials Limited and Lucas Interiors Limited as appropriate. All requests for technical support and maintenance are routed through MSBC's Product Support Desk in Kings House, Reading.

7 Benefits of using the pod

Since the construction of the first pod in Kings House, Reading, in March 1985 and the second one in the International Management Centre at Bridge House, Putney, many senior ICL managers have held and attended meetings in a pod and senior managers from about a hundred major industrial, commercial, financial and governmental organisations have visited a pod; indeed, several major ICL customers have used the pod for major internal meetings, with MSBC staff providing support before and during the meeting.

On first visiting the pod, the majority of people are immediately impressed by the range of technology available and by the ease with which it is controlled. After spending a couple of hour inside it, they begin to appreciate the success of the pod as an environment in which the participants in a meeting are encouraged to behave more effectively than they might in a conventional meeting room. A considered assessment of the effectiveness of the pod must, of course, be based on extended experience of it. A number of ICL managers have now accumulated such experience, and their conclusions are:

– that tightly regulated review meetings take less time

- that in the majority of review meetings, which are structured but not tightly regulated, the emphasis shifts from argument about the problems to discussion of the solutions
- that in unstructured or semi-structured meetings, such as when brainstorming or other using creative techniques, the creativity of the participants is enhanced
- that the pod provides an ideal invironment for Decision Conferencing.

Since it is widely recognised that senior management decisions are those with the 'highest value', that is they have the greatest potential financial and organisational impact, the greatest benefit will accrue to an organisation owning a pod if it is used for making senior management decisions. However, it is recognised that a wide variety of groups, including project management teams and software design teams, can benefit from using the pod. A number of ICL senior managers are now convinced that the pod represents a major advance in the application of information technology to business problems and see that this is due not only to the ease with which the technology is integrated and controlled but also to the behavioural context within it is used. Academia sees the pod as the first proper application of group behavioural thinking to the problem of providing IT-based assistance to senior management deliberations.

8 Concluding remarks

The pod represents a novel collaboration between ICL, an Information Technology company, Lucas Interiors, a furniture and interior design company, and Audio Visual Materials, an audio-visual systems company, to apply the results of research in a number of apparently unrelated academic and business disciplines to supporting senior managers in their group activities. It is a product totally unlike anything that ICL has produced before – 'the first ICL product that you sit in, not at' – and its very success suggests that major opportunities are open to any technologically-oriented company that is prepared to exploit the results of research in areas not conventionally regarded as relevant to its own industry.

Managing change and gaining corporate commitment

Peter Hall
ICL Management Support Business Centre, Reading, Berkshire

Abstract

This paper describes one of the processes which ICL has evolved over the last few years to support the programme of changing its "organisational capability".

It explains the need to change from strategic planning to a strategic management philosophy and the need for mangement support tools that are simple enough to be effective and yet flexible enough to have wide applibability.

One such process is Decision Conferencing. The description includes three case studies and several examples of applications that have been so successful inside ICL.

This process is now available as a product for use by ICL's customders. Initial reaction has been outstandingly good and there is a growing evidence of a rapid build up of demand.

1 Introduction

In most companies in the highly turbulent business world of today the main criterion for success is to have an organisational capability that is flexible enough to manage necessary change yet stable enough to be efficient. Decision Conferencing is one of the tools created by ICL to help meet this need. It was developed in collaboration with the Decision Analysis Unit of the London School of Economics, originally for use within ICL. Its essential purpose is to help a management group to reach a shared understanding of a problem and to gain commitment to action.

There is a growing awareness in the field of management science that creating a strategy for a company, though important, is relatively easy, but 'making it happen' is what seems to defeat most management teams. Company strategy may be all about change and how to bring it about in a controlled way, but resistance to change can be very strong – often for good reasons.

ICL has evolved Decision Conferencing over the past few years as one of a set of tools designed to help itself to 'Manage Innovation and Change'. The background of 'Corporate Renewal' in ICL was described this year in a series of articles in the *Financial Times* by Christopher Lorenz. The success of this programme is now evidenced by the progress made towards 'Strategic Management' where line managers and their teams own both their short and long term objectives and are committed to the strategies for achieving both. Tools such as 'Mission Objectives and Strategies' and 'Strategic Framework' have also been used to help to cement a new way of doing things into the new culture of the company.

Experience in using these tools has demonstrated the need for a decision support system that is useful over a wide range of different circumstances. Every management team is different and their needs differ. These depend on the level in the organisation, the stage of development of the team, the individuals' training and background, the nature and maturity of their business, and the problem they are trying to solve.

Conventional Decision Support Systems tend to be data based, and there is no data about the future. They are relatively inflexible (apart from 'what ifs' within a defined structure which hides the real assumptions) and do not allow consideration of different perspectives.

2. Decision Conferencing

It is to meet this need for flexibility that Decision Conferencing was created. As every management group is different, so every Decision Conference is different. It is only by experiencing the process that one can fully understand the beneficial effects of preference technology.

The purpose of Decision Conferencing is to help a management group to reach a shared understanding of a problem and gain a commitment to action.

2.1 What is it?

Decision Conferencing is a two-day problem-solving session attended by 'owners' of a problem. Differing perspectives of the participants are combined in one computer model that is generated on the spot, interactively with the group. By examining the implications of the model, then changing it and trying out different assumptions, participants deepen their understanding of the problem and are helped to reach agreement about what to do.

The process is guided by three supporting staff*:–

> the *facilitator*, who is experienced in group processes and decision technology, helps the group to focus on the task, identify the issues, model the problem and interpret the results.

*The supporting staff are not specialists in the subject matter of the conference and do not act as mangement consultants. Their role is to help the group structure their thinking about a problem, develop new insights about it, and arrive at a shared understanding.

the *analyst*, who attends to the computer modelling and helps the facilitator.

the *recorder*, who uses a projected word processor to help the group agree and own the words that describe critical issues, to record the thinking as it happens, and to provide the team with documentation to take away with them.

2.2 Stages in a Decision Conference

After an initial introduction by the facilitator the group is asked to discuss the issues and concerns that are to be the subject of the two days. An attempt is made to formulate the nature of the problem. Does the group need to generate a new strategic direction, or is strategy itself the issue? Perhaps budget items or projects require prioritising. Evaluating alternative plans, ventures, systems, bids or projects may be required, especially if objectives conflict.

Once the problem has been formulated, work begins on constructing the model. This is usually an abbreviated, though not simplistic, representation of the group's thinking about the problem, with the form of the model drawn from decision theory and techniques, and the content contributed by the participants. Some examples of the model types are given below. Both data and subjective judgements are then added to the model between and across the options. The computer output is then projected onto a large screen so all participants can see the results.

These initial results are rarely accepted by the group, but the model is flexible enough to allow modifications suggested by participants, and different judgements to be tested. Many sensitivity analyses are carried out, and gradually intuitions change and sharpen as the model goes through successive stages. Eventually, this process of change stabilises, the model has served its purpose, and the group can turn to summarising the key issues and conclusions. An action plan is then created so that when participants return to work the next day they can begin to implement the solution to the problem.

2.3 Decision Conference rules

Experience with Decision Conferencing has shown that following these four rules contributes to the success of the sessions:–

1 The chief decision maker must be present. This person's perspective on the problem as it develops is important to the Decision Conference. If he/she is absent the recommendations of the group are often rejected by the decision maker who feels that the group did not consider certain crucial factors

2 All major problem owners must attend. A problem owner is anyone whose perspective on the problem can contribute usefully to its solution. Usually, between 8 and 12 people will adequately represent all perspectives. It is by exploring these different perspectives with the model that a shared understanding of the problem can emerge (though that does not mean that everyone has to agree about everything!)

3 No papers or printed material may be consulted during the problem formulation stage. It is people who perceive problems, usually by sensing that a discrepancy exists between some desired goal and the current state of affairs. Human judgement can best formulate the nature of a problem; printed materials often distract participants from the real problem.

4 The problem must be a live one. Using Decision Conferencing to justify decision already taken doesn't work because hindsight biases prevent the development of an agreed model. For hypothetical problems, judgements are too ill-formed to give meaningful results.

2.4 Benefits

Organisations that have used Decision Conferencing say that the service helped them to arrive at a better and more acceptable solution than they would have achieved using their usual procedures. Agreement was reached much more quickly. Many Decision Conferences have broken down barriers created by lack of consensus, the complexity of the problem, vagueness and conflict of goals, and failure to think creatively and freshly about the problem.

The model developed in a Decision Conference lends structure to thinking, and so allows all perspectives on a problem to be represented and discussed. The process facilitates communication among participants, providing 'a way to talk differently', as one person put it. It surfaces assumptions that are often different from one person to the next. Because the model developed by the group shows what the organisation can do, rather than just describing what it does, creative and lateral thinking is encouraged. Overall, Decision Conferencing helps a shared understanding to emerge from differences in perspective, it builds commitment and generates action plans.

2.5 Foundation of Decision Conferencing

Decision Conferencing draws on experience and research from three disciplines: decision theory, group processes and information technology. Decision theory is brought to bear in the development of the model, ensuring its internal consistency, so that subsequent changes to one part of the model do not require alterations to other parts that are considered satisfactory.

Consistency is an important feature, for it allows several models, possibly developed by different parts of the organisation at separate Decision Conferences, to be combined, without alteration, into one overall model representing a higher organisational perspective.

Research on group processes has identified conditions and situations that may cause a group to become diverted from the task, engaging in behaviour that does not contribute to creative and effective problem solving. Knowledge of group processes is used by the supporting staff to help the group stay oriented to the task.

Finally, information technology in the form of a computer, computer programs and a projection system allows the model created by the group to be implemented on the spot, and provides the means for immediately showing the results. Thus, participants can try out different judgements, see the results without delay, and so modify their views or the model until a satisfactory representation of the problem is obtained. The programs used for Decision Conferencing have been created specially for this purpose, and have evolved over a period of years as experience was gained. The displays are mostly pictorial and graphical, and facilitate easy interpretation of results.

3 Case studies

Because every Decision Conference is different the above description cannot do more than set out the underlying principles. The following case studies, drawn from some of the management problems we have tackled so far, will give more insight.

Every Decision Conference is confidential to the group members. It is important that individuals feel totally free to say what they feel. This enables the group dynamics to work in surfacing the real assumptions and issues. Therefore, none of the case studies are attributable, but they do serve to demonstrate the power of the process in different circumstances.

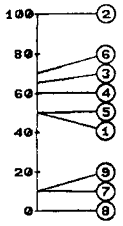
3.1 Strategy creation

In this case the business centre was well along the track with their objectives, but now needed to agree their strategic direction in the face of a very uncertain future market place.

Their view at the start was that their choice of strategy depended on which future view they took. We spent the first part of the Decision Conference agreeing alternative future scenarios of their business arena.

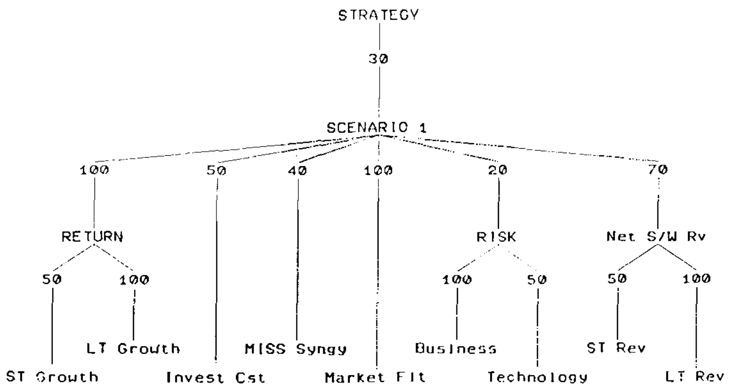
These were described in detail using a word processor projected onto a large screen. Every member of the group shared in the generation of the descriptions and all their views on possible futures were included in one of the scenarios.

They then considered a wide range of alternative strategies that they might adopt and eventually agreed nine options which were also described in detail. In addition, they reconfirmed their mission and objectives and with these in mind chose nine criteria for assessing which options they preferred. Then by considering only one criterion at a time they were able to agree by pairwise comparison the ordering of their preferences for the options. Eventually the options were positioned on a 0–100 preference scale.



For instance, for short term growth option 2 is most preferred at 100 and option 8 least preferred at 0. The discussion and debates in reaching agreement on the ordering gave new insights into the surrounding issues and caused a redefinition of the problem and the options.

Finally the group agreed the relative importance of each criterion and established the weights as in Fig. 1.*



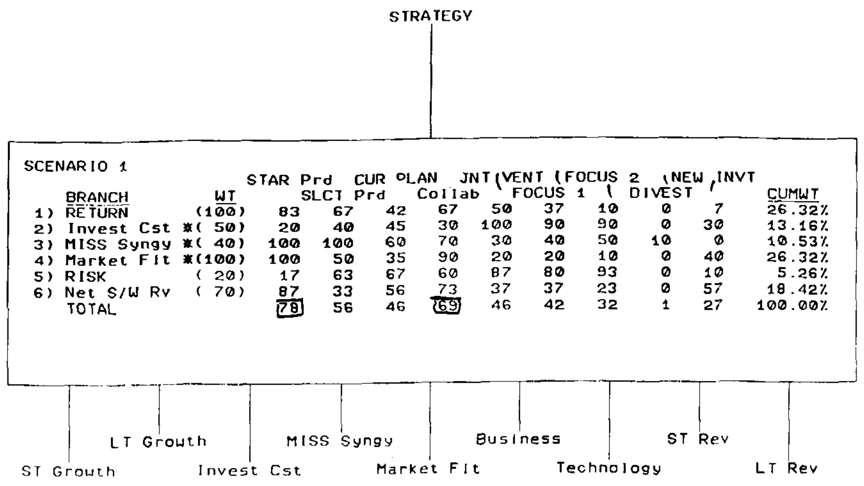
OPTIONS	ST Growth	LT Growth	Invest Cst	MISS Syngy	Market Fit	Business	Technology	ST Rev	LT Rev
1 STAR Prd	50	100	20	100	100	10	30	60	100
2 SLCT Prd	100	50	40	100	50	60	70	20	40
3 CUR PLAN	65	30	45	61	35	50	100	100	35
4 Collab	60	70	30	70	90	60	60	60	80
5 JNT VENT	50	50	100	30	20	100	60	70	20
6 FOCUS 1	70	20	90	40	20	80	80	70	20
7 FOCUS 2	10	10	90	50	10	100	80	50	10
8 DIVEST	0	0	0	10	0	0	0	0	0
9 NEW INVT	10	5	30	0	40	10	10	10	80

Fig. 1

This structure, the scores and the weights were then input to the computer model which multiplied the scores by the weights to give a single value for each option.

For instance, in Scenario 1, option 1 was preferred with a value of 78, to option 2 on 69, etc. (as in Fig. 2 below).

*The Figures are exact reproductions of what is displayed on the screen during the session.



```
SCENARIO 1
                          STAR Prd  CUR PLAN  JNT(VENT (FOCUS 2  (NEW ,INVT
   BRANCH         WT         SLCT Prd   Collab   FOCUS 1  \ DIVEST '        CUMWT
1) RETURN       (100)    83    67    42    67    50    37    10    0    7    26.32%
2) Invest Cst  *( 50)    20    40    45    30   100    90    90    0   30    13.16%
3) MISS Syngy  *( 40)   100   100    60    70    30    40    50   10    0    10.53%
4) Market Fit  *(100)   100    50    35    90    20    20    10    0   40    26.32%
5) RISK         ( 20)    17    63    67    60    87    80    93    0   10     5.26%
6) Net S/W Rv   ( 70)    87    33    56    73    37    37    23    0   57    18.42%
   TOTAL                 78    56    46    69    46    42    32    1   27   100.00%
```

Fig. 2

The group then repeated this process for the other scenarios generating different scores and weights for each. The first pass results were then obtained by putting weights on each scenario, judged to be the possiblity of it happening. However, even more insight came from the graphical plot in Fig. 3.

This shows that the choice does depend on which scenario occurs. In scenario 1 option 1 is preferred. In Scenario 2 option 2 is preferred. But 1, 2 and 4 all dominate any of the other options in both scenarios.

The group was able to agree that in either scenario options 8, 9, 7, 3, 5 & 6 would not be considered. The focussed solution would be one that combined options 1, 2 & 4.

The model offers facilities which allow the group to explore their own individual beliefs and examine them against those of the rest of the group. For example, a belief that it is all about the importance placed on risk, can be explored in Fig. 4.

The solution is dependent on that weight, but you would have to treble the weight before it made a difference. They could all agree that this did not seem reasonable.

This process of exploration enabled the entire group to understand the relative importance of individual members' beliefs. This helped to break down barriers and, in turn, make way for acceptance of a 'group view'.

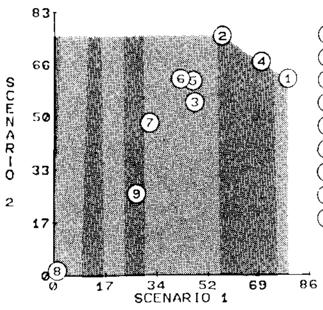


Fig. 3

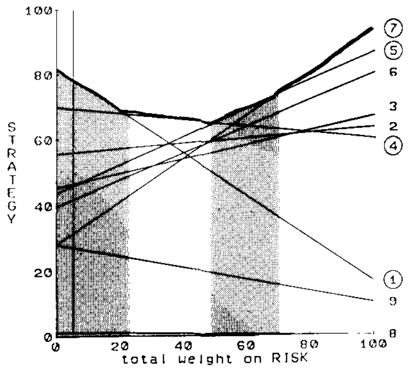
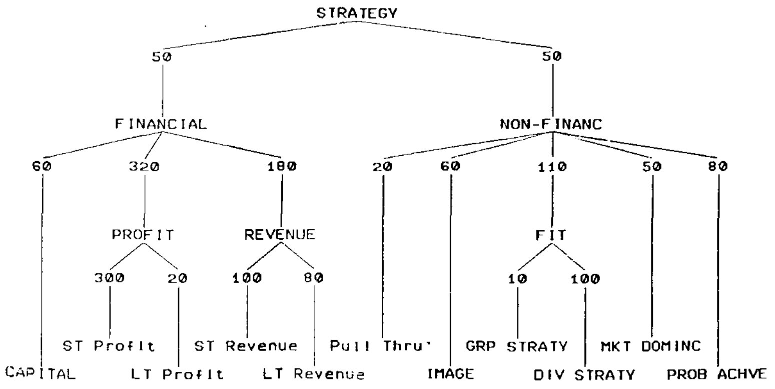


Fig. 4

3.2 *Business strategy*

This case was to decide on the business strategy for an extremely successful new business initiative which now needed more investment. The initial discussion revealed that there was a difference of opinion in the team as to how to gain best advantage. Part of the team felt that they should complete the strategy they had started two years earlier, and maintain their ownership and control. Another part felt that they should change to an outward going joint venture and get early return with which to fund future growth. The argument was used that what was best financially for the team did not fit with the total company strategy. We therefore modelled the possible business options and assessed then against both hard financial criteria (Revenue and Profit forecasts) and non-financial softer criteria (Fig. 5).



	CAPITAL	ST Profit	LT Profit	ST Revenue	LT Revenue	Pull Thru'	IMAGE	GRP STRATY	DIV STRATY	MKT DOMINC	PROB ACHVE
PULL OUT	33	0	0	0	0	100	0	100	0	0	0
SELL OFF	83	91	10	0	0	0	25	0	10	0	80
SELL 1/2	66	89	40	0	0	50	80	60	50	10	60
DIV PART	17	77	90	0	0	20	60	100	35	10	80
CUR PLAN	0	58	60	0	0	20	55	90	30	10	100
3+ FOCUS	100	95	50	0	0	55	80	60	50	10	20
JNT VENT	0	83	100	50	100	50	70	70	80	100	40
3 and 7	66	100	50	50	100	100	100	75	100	100	30
3+ MKTNG	66	68	50	100	50	85	90	80	60	50	35

Fig. 5

The discussion around the model gradually moved both parties to change their views as they reached a better understanding of each other's perception of the problem in these terms. In particular, technical ownership and business exploitation became aligned. They were able to generate new options and in the final model one strategy was clearly preferred for both financial and non-financial reasons (Fig. 6).

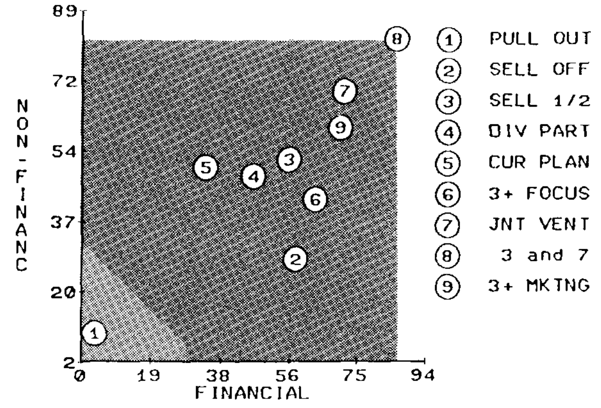


Fig. 6

The group then went on to define a precise strategic direction in a very complex area, and created an action plan which has now come to fruition. Two months after the event, they commented that:

"*It was the turning point for our group. It changed the mindset of our management team in a very constructive way and we are now following the new strategy very closely.*"

3.3 Sales resource allocation

A different model which has also proved very beneficial is designed to help a management team with trade-offs between parts of the business. In this case a sales management team was operating in seven countries in which each country manager had optimised the use of his limited resources. More rapid growth, however, was eluding them and they wanted to see how to improve the use of the total resource. Each country manager was asked to prepare alternative plans with resource levels greater and less than the current plan for the next three years.

During the initial discussion, it became clear that there existed the usual budgetting attitude of asking for more than you hope to get. There was also a lack of understanding of each other's businesses in widely differing business environments.

This was modelled by first considering the costs of possible practical levels of investment in each country.

Then as a group, they assessed their performance on a 0–100 scale for each options against four benefit criteria (profit and revenue over the next three years, the risk of achieving it and the long term potential of the business after the three years). Fig. 7 shows an example for one of the countries.

VARIABLE 1: GREECE

		COST				BENEFIT				
		OPEX	SERV	CAPEX	TOTAL	PRFT	REV	RISK	POT'L	TOTAL
1	Care & Maintenance	10	30	0	40	0	0	100	0	0
2	STATUS QUO	195	105	15	306	10	10	75	15	7
3	Larger Branch	375	120	30	507	25	30	15	25	6
4	+ Local Assembly	465	210	45	693	45	50	45	70	56
5	+ New Product	765	240	145	1063	70	75	20	90	79
6	+ Government Sector	1125	270	145	1453	100	100	0	100	100
	WITHIN CRITERION WTS					29	57	40	70	98
	ACROSS CRITERIA WTS	100	100	40		100	50	60	50	

Fig. 7

Looking at one criterion at a time, the group than had to assess the relative importance of the difference between the best and the worst case between each of the countries. Finally, they weighted the importance of the criterion.

These weights are then used in the model to produce a single benefit value and a single cost value for each option in each country. There are literally thousands of different combinations of total investment that could be chosen.

The computer starts with the lowest level in each country (similar to zero base budgetting) and looks for the highest benefit/lowest cost in all these combinations. It then looks for the next highest and so on. This process produces an 'order of best packages' (Fig. 8) where successive increases in investment result in relatively smaller increases in benefit.

*		VARIABLE	ORDER OF BUY		COST	CUM	CUM %
			LEVEL			COST	BENEFIT
0	1	GREECE	1	Care & Maintenance	40.0	40.0	0
0	2	YUGOSLAVIA	1	Agents Only	50.0	90.0	29
0	3	USSR	1	Agents Only	45.0	135.0	29
0	4	POLAND	1	Care and Maintenance	360.8	495.8	29
0	5	CSSR	1	Cutback	575.0	1070.8	29
0	6	HUNGARY	1	Source Mrkt from UK	390.0	1460.8	29
0	7	BULGARIA	1	Close Down	40.0	1500.8	29
1	7	BULGARIA	4	+ Jnt Systems Centre	559.0	2059.8	229
2	5	CSSR	3	+ Local Countertrade	709.0	2768.8	329
3	6	HUNGARY	3	+ Jnt Systems Centre	280.0	3048.8	368
4	2	YUGOSLAVIA	5	+ Joint Venture	1850.0	4898.8	616
5	4	POLAND	4	New Strategies	743.2	5642.0	716
6	3	USSR	3	+ Re-Engineering	370.0	6012.0	765
7	4	POLAND	5	Joint Venture	378.0	6390.0	810
8	1	GREECE	4	+ Local Assembly	653.0	7043.0	866
9	5	CSSR	4	+ UK Counter Trade	150.0	7193.0	878
10	6	HUNGARY	5	+ Jnt Prod. Devlpmnt	313.0	7506.0	900
11	1	GREECE	5	+ New Product	370.0	7876.0	922
12	1	GREECE	6	+ Government Sector	390.0	8266.0	943
13	3	USSR	5	+ Re-Engineering BIG	620.0	8886.0	975
14	5	CSSR	5	+ Manufacturing Coop	351.0	9237.0	991
15	3	USSR	6	+ Jnt Develpmnt Cntr	295.0	9532.0	999
16	3	USSR	7	+ New Product Range	445.0	9977.0	1000

Fig. 8

This can be viewed as a curve in Fig. 9. In the shaded area are all the possibilities the group had thought of (but there can of course be no points above the curve). In particular there is one point **P**, the current plan or status quo in each country.

But there is a better point **B** which costs about the same but gives much more benefit.

Alternatively there is a cheaper point **C** which can give the same benefit for considerably less cost. Fig. 10 shows how to reach **B** or **C** from **P**: you should move to the lowest level in three of the countries and use that resource to invest more heavily in the other four.

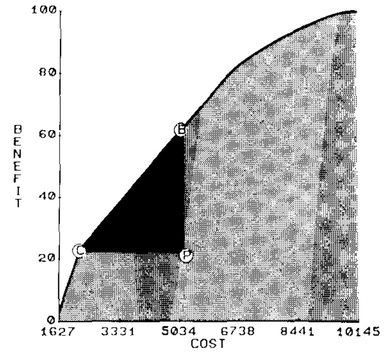


Fig. 9

VARIABLE	LEVEL 1	2	3	4	5	6	7
1 GREECE	CB	P					
2 YUGOSLAVIA	C		P		B		
3 USSR	CB	P					
4 POLAND	CB	P					
5 CSSR	C	P	B				
6 HUNGARY	C	P	B				
7 BULGARIA		P		CB			

Fig. 10

The country managers realized that the best possible plan ('**B**') for the group as a whole was not the sum of the individual country plans ('**P**'). Many new ideas were generated. The model was used to try these out and helped to cross communicate the fundamentals of each others businesses. Eventually they could agree that the structure, their preferences and the weights were adequate to represent what they should do. Even the country managers that were putting their patch onto care and maintenanace only bought into this group view.

They were then able to go on and agree the level of operation, and to set the objectives, strategies and measurable milestones for each of the countries. They identified the actions they needed to take over the next few months.

In fact the result was used as the budget for the following year and resulted in doubling the revenue and trebling profit of the group as a whole.

4 Applications in Decision Conferencing

The three case studies just described were aimed at explaining the kind of models available as part of the process. The applicability of the process is of course much wider, limited only by the capability of the management team themselves.

Within ICL we have now done over 50 conferences over a wide spread of applications and levels in the organisation, with outstanding success. The measure of success is the management team's perception of whether they reached a shared understanding and achieved real commitment to action, both after the event and a few months later with hindsight.

4.1 Development spend

At the board level the issue was which development activities to undertake to support the company's strategic direction given that there is a limit on the total research and development spend. The company had struggled with this problem for years and used various different techniques. The bids always exceeded the available funds and projects had to be cancelled, cut back or delayed to allow for necessary new starts. However good the supporting business cases the ultimate decision is always a trade-off between different areas of business activity.

The resource allocation model fits this problem well. The choice is between different levels of spend in each development area against the criteria of how best to achieve the company's objectives. The order of best packages now gives a prioritised list of devlopment projects indicating which projects should be brought in if more funds become available and which should be looked at first in the event of a cut back. But most of all it has developed a common way of thinking about development decisions, and the slope of the benefit/cost curve gives a standard by which new ideas can be judged.

Similar models with similar criteria were also developed in other Decision Conferences, at Divisional, Business Centre, and Business Unit levels with great effect both on the consistency of the development approach, and with the commitment of the management team to a new corporate direction.

4.2 Cross business strategies

Contrast with another problem area of how to develop a new stategy that was an essential ingredient to the company's total future. This new strategy would not be supported by any of the current business strategies on their own; only collectively. Here the problem is closer to the Business Strategy case study, but with the need to get the commitment of people from different management teams to co-operate for the common good.

4.3 Sales divisions and countries

Many countries and sales areas found the process helpful both in creating and cementing a strategic direction, and in resource allocation. Where appropriate it was sometimes possible to build a strategic direction model on the first day, and a resource allocation on the second, or even extend the event to three days to allow this. The important thing is to leave enough time at the end to convert what is agreed in the model world back into an action plan in the real world and still keep the same level of commitment to it.

Where the indigenous language was not English it was interesting to see how Decision Conferencing helped to create a symbolic language. This allowed complete communication for the first time, and built a shared understanding that transcended the language barriers into the future

4.4 Communications of preferences

In general the models created by a management team have served their purpose in acting as a communication tool within the group. But in some cases additional use has been made of the model:

1 To enable the group to communicate their preferences to other management teams. For example:

> a Development Division used the model to show the Sales Divisions which development projects they had selected and why;
>
> and, two management teams used the model to explore what they thought were conflicting objectives and negotiate a solution to which they could both agree

2 As an ongoing tool to provide a top level guide to more detailed business plans. This is essential in keeping live the understanding reached and communicating the rationale internally down through the group.

4.5 Creating new organisational identity

Another way of using Decision Conferencing is to help build a management team's capability over several years by assimilating the process into their culture so that it is a normal way of tackling problems as their business develops.

For instance, the Retail Business Centre used it over a period of three years:

> firstly, to establish their Mission Objectives and Strategic Direction;
>
> secondly, to put in place their supporting organisation and systems;
>
> and finally, as the way to set their budget.

Network Systems Division have also done several since they were formed. These were aimed at not only establishing their strategy, but also building a creative, effective team that could achieve the results needed during the start up phase.

Outside ICL, Decision Conferencing has proved very beneficial in several other companies in the U.K., including B.O.C., Frizell, Mars, Pactel and Dollond & Aitchison. A wide variety of business problems has been addressed. Examples are:

> the choice of new product launch;
> establishing an Information Technology Strategy;
> deciding between investment opportunities;
> establishing corporate international strategy;
> and many others.

Collaborative research

There are three other units on a worldwide basis who are using Decision Conferencing. In the U.S., Cam Peterson of Decision Conferences, Inc. has been working with Westinghouse and several defence establishments. Professor John Rohrbaugh of New York State University has been working with the public sector in New York State. In Hungary, Janos Vicsenyi has been using a similar process and has recently signed a collaborative agreement with ICl and Szamalk to establish a Decision Conferencing centre in Budapest.

On a worldwide basis we now have a database of some 200 events where senior management teams have been addressing top level problems in two day working sessions. The University Research Council have funded a three year programme of 'research into high level decision support through the analysis of Decision Conferencing.' A network of people working and researching in this field in universities and business schools has now been established with an annual conference at Minster Lovell, near Oxford. There is a growing awareness in the U.S. of the need for "Group Decision Support" (G.D.S.). Several companies have started to bring in products aimed at extending decision support from the individual to the group. However as most of them are starting from conventional database systems as yet no-one else appears to be using decision theory and modelling of preferences in the same way.

In collaboration with the LSE we are moving ahead to develop other tools and techniques which will build on this competitive advantage and be available in 1987. There is another product called "MAUD" (Multi Attribute Utility Decomposition) available now which is aimed at supporting an individual decison maker on a Quattro PC (and on OPD by the end of the year).

7 Conclusion

Decision Conferencing is an excellent example of a product which we have developed internally to help our own managers to be effective strategic, as well as operational managers. It is now available as products in support of the company mission to improve our customers management effectiveness.

References

1 TOM GILB.: Design by Objectives.
2 PETER HALL.: The Way to Set Objectives.: The ICL Way — MSBC.
3 ELLIOT JACQUES.: The General Theory of Bureaucracy, Heinemann 1981.
4 PETERS and WATERMAN.: In Search of Excellence, Harper & Row 1982.
5 GOLDSMITH and CLUTTERBUCK.: The Winning Streak, Weidenfeld & Nicholson 1984.
6 MICHAEL PORTER.: Competitive Strategy, Macmillan 1980.
7 JOHN CHANDLER and PAUL COCKLE.: Techniques of Scenario Planning.
8 LARRY PHILLIPS.: Decision Support for Senior Executives. }
9 LARRY PHILLIPS.: Decision Analysis and the Application in Industry. } DAU LSE
10 LARRY PHILLIPS.: The Theory of Requisite Decision Conferencing. }

An approach to information technology planning

S.J. Pollard
ICL Group Information Services, Putney, London
C.R. Crawford
ICL United Kingdom Limited, Putney, London

Abstract

This paper outlines an approach to Information Technology planning which is directly linked to the objectives and strategies of the subject enterprise. It defines a set of prime management strategies for IT from which a selection must be made for sub-sets of the enterprise. These choices lead to the further selection of support strategies for IT and these, in turn, help determine the essential functionality of applications and their relative priorities. The paper includes a planning technique which will help to ensure that the application development process does not lose sight of these identified business needs.

1 Introduction

'Successful Information Technology is that which directly supports the strategic objectives of the enterprise – unsuccessful IT is that which does not.'

Like all motherhood statements, this suffers from the fact that it is both obviously true and normally forgotten in the consideration of detailed requirements.

The purpose of an Information Technology (IT) strategy is to provide control by identifying the corporate strategies and determining how IT contributes to them. The resultant IT strategy, in turn, defines the objectives, methodologies, management approach and funding mechanisms for IT within the enterprise. Note that we use 'information technology' as an umbrella term for all IT elements – hardware, system software, application software, networks and so on.

This paper specifies an approach to IT planning which is primarily driven by the strategies and objectives of the subject enterprise, rather than by the technology. The word 'approach' has been used because this document does not claim to provide a completely prescriptive methodology for strategic IT planning.

The paper endeavours to provide some insight into the inputs, processes and outputs of the planning activity but the state-of-the-art does not permit these to be precisely defined. Planning consultants may take comfort from the fact that 'IT planning-by-numbers' is not a tenable proposition and that their skills will therefore remain in demand for the foreseeable future.

This paper is a synthesis of a number of published papers and books dealing with aspects of the strategic deployment of information technology. We have endeavoured to combine, to modify and to add to these individual notions to create an implementable approach. While acknowledging our sources, albeit anonymously, we remind the authors that:

'To copy one person's work is plagiarism but to copy two or more is research.'

2 A pictorial overview of the approach

Our starting point is an elementary model of the business planning process which, hopefully, is non-controversial. We use it to introduce the richer picture which follows:

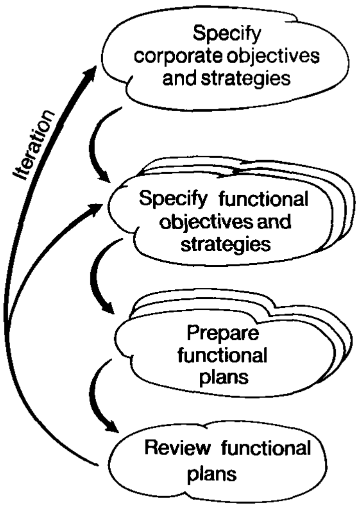

Specify corporate objectives and strategies
Iteration
Specify functional objectives and strategies
Prepare functional plans
Review functional plans

Fig. 1 Elementary Planning Model

The model highlights that planning is essentially both hierarchical and iterative in nature.

Overlaid on this simplistic model is further detail to reveal the 'rich picture'. The body of the paper is devoted to an explanation of this view but there are a few conventions which should be explained at this point.

Processes are contained within 'clouds'. This imagery is intended to reinforce the sentiments expressed above that planning processes are not mechanistic but demand vision, business awareness and intellect.

Inputs and outputs are shown, conventionally, within boxes while the 'mesh effect' denotes a part of the process where an idealized requirement is 'filtered' to transform it into a real world view.

Finally, for simplicity, we have avoided making explicit every possible iteration and have indicated major loops only.

3 IT considerations within corporate-level planning

3.1 Introduction

To be effective, IT considerations must occur early in the business planning cycle. This does not imply that IT is necessarily always of paramount importance but that its deployment or otherwise should be a conscious decision at corporate level.

This document is not about corporate business planning per se but it is the essential starting point.

3.2 Corporate objectives and strategy

The key to an IT strategy is a clear statement of overall corporate objectives and strategy.

For the purpose of IT strategy, this is defined as the way in which the company intends to compete in its market place – against its competitors, suppliers and new entrants – to capture and retain its prime target customers.

The key strategy is its competitive strategy and a simplistic view of this is contained in three prime expressions:

- A strategy of lowest cost supplier across the whole industry.
- A strategy of product differentiation across the whole industry.
- A strategy of targeting specific customer/product niches within the industry.

While any large organization will tend to utilise all of these strategies over time, and indeed simultaneously, it is important to establish both the currently predominant strategy – and any intended change both for the specific company and across its industry – since the IT strategies and the nature of the IS applications to support these broad competitive strategies are radically different.

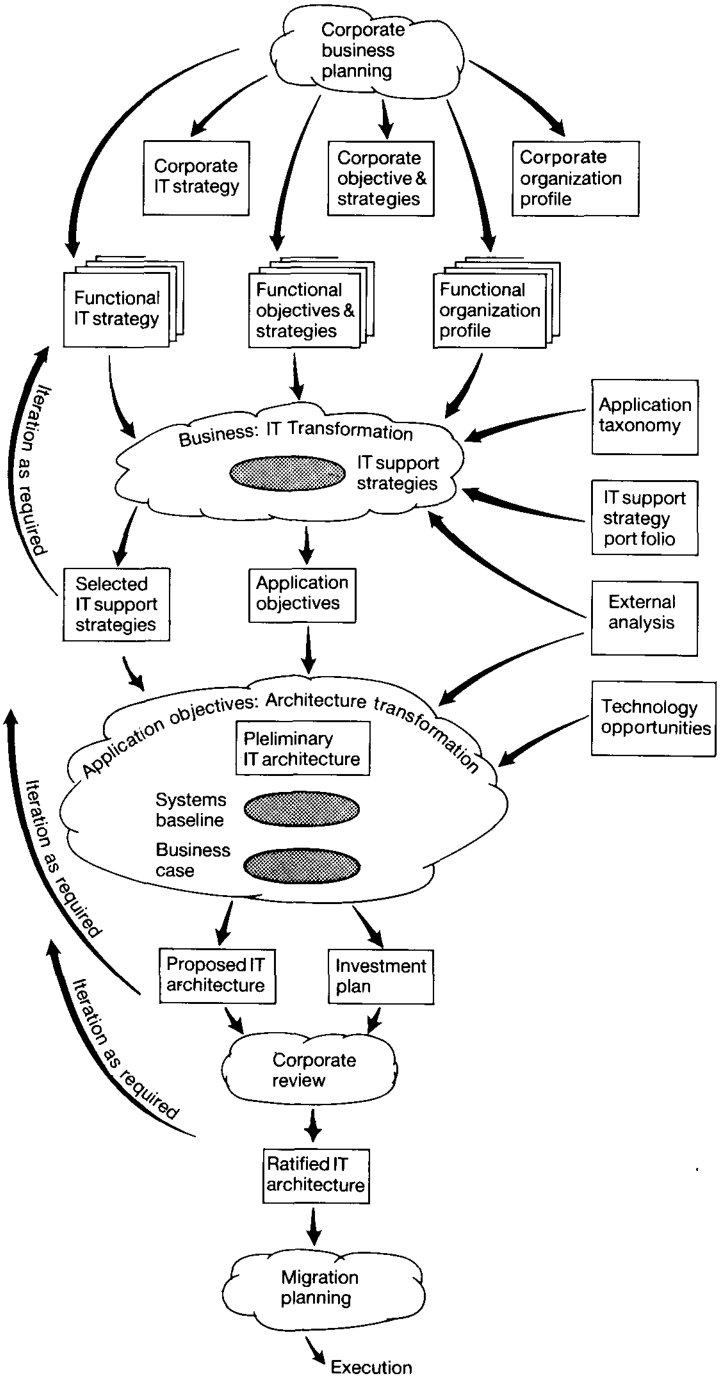


Fig. 2 A pictorial summary of the approach

It is self-evident (but not always rigorously observed) that the corporate objectives and strategies must be clearly expressed, must be logically consistent and must be demonstrably loyal to a declared definition of terms (see below). It is not the primary role of the IT planner to question the veracity of such items which are logically constructed but it is imperative that the planner does not proceed on shaky foundations. Quality at this stage is vital.

A Mission statement provides a useful 'summary' of the enterprise and the authors' preference is for this to be created as a (Checkland) 'Root Definition'. This will help to yield a Mission statement which is concise and comprehensive rather than one comprised of linked buzz-phrases.

Finally, in this section, we offer some short definitions of the terms employed above. The collective is given the acronym MOSC – Mission, Objectives, Strategies and CSFs – and we feel the religious overtones of this acronym to be entirely appropriate.

- *Mission:* This is a pithy statement which encapsulates the complete enterprise. It should make clear: the customer, the actors, the transformation, the perspective, the owners and the environment. (The terms are Professor Checkland's and the reader is referred to his work for a more extensive explanation.)
- *Objective:* A defined state to be achieved in a defined timescale. In essence, an objective is a 'What?' by 'When?' statement. It is essential that the 'What?' must be measurable, that is, the achievement of the desired state must be unambiguous.
- *Strategy:* The means by which objectives are achieved – those general courses of action to be pursued in achieving Company objectives. In essence, strategies should embody 'How?' statements. Strategies cannot exist in isolation but must be directly linked to objectives; there may be more than one strategy per objective.
- *Critical Success Factors:* CSFs are 'the key areas of the business [organisational entity] in which high performance is essential if objectives are to be met'. As this implies, CSFs need to be strongly linked to objectives and strategies to ensure coherence. They provide a useful check-and-balance in addition to the value they add in their own right. Measures need to be established for each CSF so that deviations may be detected.

3.3 Prime IT strategies

Prime IT strategies can also be conveniently compartmentalized into four, of which two are reasonably well researched and understood and two are not.

These are best represented by the following matrix:

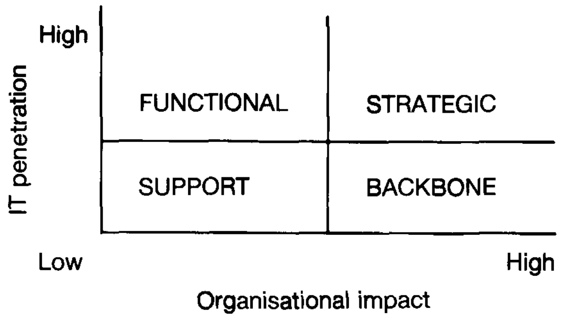


Fig. 3 Penetration/impact matrix

The axes of this matrix represent dimensions covering the nature of IT within the organisation.

IT Penetration is the extent to which hardware and applications have extended across the organisation. 'Low' penetration is best exemplified by the traditional batch machine in its DP fortress and 'High' by a proliferation of dumb and intelligent terminals throughout the company.

Organisational Impact is the extent to which the organisation is dependent on IT. At the 'Low' end the company could operate with no IT at all; at the 'High' end it is in serious danger of collapse if its systems are not functioning.

Within this matrix, IT strategies are represented by the four quadrants which are:

- Support – A support strategy is one where IT is used only where absolutely necessary, generally for centralised administrative tasks and essentially as a clerical replacement tool.
- Functional – A strategy of not using IT corporately but to support specific functional units. This is characterised by a widespread use of mini and departmental computers – with little if any exchange of data – primarily for clerical replacement and middle management support.
- Backbone – A strategy of using IT to provide/control all administrative/data communication functions within the organisation, characterised by widespread integration of applications and dependence on networking.
- Strategic – A strategy of using IT as the means by which the organization achieves its competitive strategies. This is the least well understood of the quadrants and is currently expressed principally as examples rather than by definition. It is often characterised by the external use of IT as part of the customer/supplier interface.

As has been implied by the definitions of these strategies, the importance of establishing general strategies is that the nature of the employment of IT differs radically between them. Not only does IT change, however, but the management, funding, and organisation of IT are also substantially different.

This is further explored later but, before doing so, we will add a further layer of detail termed 'Transition'.

Broadly, Transition expresses the fact that most organizations in their use of IT will – possibly accidentally – change strategy as a result of technology and/or user pressure. When this occurs, a transition state is occupied during which the applications base is acquiring the characteristics of the new strategic set while the IT organisation and management strategy are operated as though the former strategy was still applicable.

The characteristic of this period of lag is high cost, high levels of frustration/confusion and experimentation; this requires specific controls and management approach.

A more detailed representation of the matrix introduced earlier would appear as:

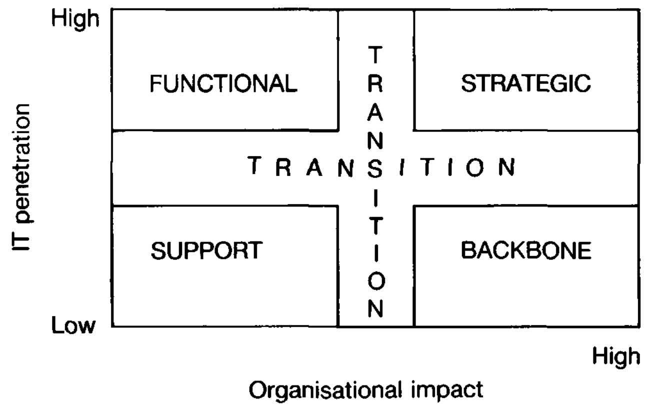


Fig. 4 Modified penetration/impact matrix

The message for the consideration of IT within corporate-level planning is clear. Firstly, the IT strategy needs to be deliberate and to act as a control factor on the IT within the organization; in addition it identifies what is not to be done as well as what is to be done. Secondly, it shows that crossing a boundary needs to be a deliberately managed action, not an accident, otherwise there is a danger of 'stalling' in the expensive position between the boundaries.

3.4 IT support strategies

These expressions of IT strategy and their implications are simplified statements. Their application depends on further detail – particularly, the qualitative differences between the four general strategies and the relationship between the competitive strategies, the IT strategy and organizational design and development.

Recognising this, the following sections are aimed at bringing together management and organizational issues in order to establish a more detailed view of IT support strategies and their influence on the nature of the application systems deemed necessary to support the enterprise. Note we use the phrase 'application system' as an umbrella term for all types of application. (These different types are explored in more detail later.)

The '*Support*' approach is primarily financially based and is a deliberately negative policy designed to constrain/reduce IT expenditure. It can occur after a highly successful innovative IT development programme but is most typically used in an organization with primitive IT usage.

The adoption of the Support strategy by an organization implies that either as a result of careful thought – or on a 'cultural' basis – the organization has concluded that IT will not contribute directly to its competitive strategies. As a result, the primary role is to reduce the corporate cost base, either by using IT to reduce labour cost and/or by reducing the IT budget.

There are two primary management techniques for implementing this strategy and, additionally, they point to some underlying characteristics of supporting applications:

- 'Necessary Evil' – Broadly this approach requires the absolute minimum of expenditure of money and management resource on IT issues, the only justification for the use of IT being that there is no other economic way of performing the task. Within this environment, IT is managed on the basis of 'Zero Base' budgets with minimal user input. Development/enhancement programmes are justified exclusively on direct monetary ROI, are based on trusted technology and will be aimed at labour-intensive, high-cost functions.
- 'Scarce Resource' – A softer version of 'Necessary Evil'. Spend and resource are above the minimum but corporately constrained to a perceived appropriate level. Within this maximum, existing IT is supported but new development and enhancement programmes are prioritised by monetary ROI and user pressure on Corporate management. Use of experimental applications and new technologies is minimised.
- 'Cost-Reduction Focused Applications' – Both of these techniques cause applications to be concentrated on cost reduction/efficiency, principally by clerical replacement at the process worker level. They actively prevent Corporate benefits being defined.

The '*Functional*' strategy is corporately neutral. IT is not expected to support directly corporate strategies but is available to units of the enterprise if appropriate.

The primary management technique for implementing this strategy, and the implied application characteristics, are:

- 'Free Market' – MIS units operate as internal profit centres and are corporately measured on this basis. User units operate to agreed budgets/targets and are accountable for ROI on their total spend, of which IT is an element. User units may acquire IT from internal or external sources, as appropriate. ['MIS', Management Information Services, is the term we use for the internal IS unit of the enterprise.]
- 'Functionally Focused Applications' – Applications are bounded by departmental and budget domains, with minimal cross-domain application/data usage. They are primarily aimed at improving budgetary control and/or cost base reduction. Finally, data is managed as a User unit resource.

The '*Backbone*' approach is based on a corporate intention to utilise IT applications as its primary method of communication and control. In this mode, the corporation is concerned to ensure that its overall administration control and communication is efficient, rather than to ensure that individual unit/functions are optimised.

The management technique for controlling this strategy, and the application characteristics, are:

- 'Corporate Control' – Corporately defined functional and data models are established as the basis of the management approach and there is corporate imposition of priorities and application functionality. ROI of a particular application is defined by its contribution to the coherence of the Network, rather than by hard monetary measures. However, hard ROI is expected from the overall IT investment.
- 'Information-Supply Focused Applications' – The key application characteristics are, at the data system level, the provision of coherent and timely data to surrounding applications and also to the information system and decision support system levels. The latter in turn have, as key characteristics, the ability to transmit and utilise the data/information.

The '*Strategic*' approach is in a sense not a strategy in its own right but is a complementary addition to one of the general strategies already described. In this approach, however, there is a qualitative difference in the objectives of applications. This, in turn, requires different management and control methodologies and, as a result, it needs separate description.

The prime difference is that, in this mode, the task is to increase the quality of the performance of a specific function of the corporation relative to current and potential competitors to such an extent that competitive advantage is achieved. Examples of supporting applications might be a superlative market intelligence system which identifies opportunities many months in advance of competing organizations or an order processing system which provides attractive services to the customer, either directly or by its speed of response.

The key management task in this arena is the identification of the strategic function, the specification of objectives and functionality and the management of the development. The task is most likely to be performed by senior/corporate management with IS management providing technical resource. The ROI is expressed in terms of increased market share, other cost/benefits being largely irrelevant.

- 'Venture Capital Approach' – The management technique is that of the venture capitalist. That is, a tranche of money is allocated and one or more experimental pilots established and tested. Those that 'work' are exploited, those that do not are quickly stopped.
- 'Application Characteristics' – There are no specific characteristics for this approach but the applications are probably aimed at automation of the process management or knowledge worker functions and/or the provision of additional product services.

A '*Transition*' strategy can be deliberately employed – as opposed to entered into by accident – when an enterprise has identified its current positioning and has determined the need for a change to a chosen new position.

In this sense, it is primarily a preparatory and training task – defining and establishing the infrastructure of tools, management methodologies and IS objectives necessary for the new strategy and the convergence path between the new and old approaches.

Included in this phase, however, is likely to be some experimental systems developments aimed at testing the underlying philosophies.

3.5 Organization and establishment profile

We now turn to a consideration of the effects of organizational design and development, and to the nature and size of the 'establishment' – the people of the enterprise. Organization does not dictate a particular approach but it causes a strategy to be more or less appropriate.

Three basic forms of corporate organization are identified and these are:

- Multiple – Organizations with a number of functionally identical sub-units controlled in all aspects of their business by a strong corporate body.
- Conglomerate – A loose grouping of functionally disparate organizations with the central corporate function principally concerned with financial management.
- Concentrate – An organization with a single functional task divided into a number of sub-tasks (e.g. R&D, Manufacturing, Sales) and a corporate

function principally concerned with co-ordination, planning and monitoring.

A key strategic issue is the observed, but as yet little understood, change in the way that the middle organizational layer functions. This seems to dictate application and technology requirements which the prime strategies only partly address.

The traditional picture of an organization is a pyramid composed of three layers with approximate manpower proportions as indicated:

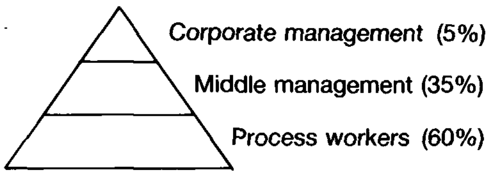


Fig. 5 Traditional organisational model

Within this general structure, control is applied and information is organized into a Tree Structure, namely:

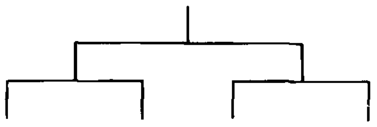


Fig. 6 Tree structure

These organizational models are fundamental assumptions used in current management processes and in considering the application of IT. However, it appears that this model is no longer entirely sound. If functional-task and data-exchange studies are made, then it appears that a number of environmental factors – the need for increased speed of response, the application of technology, cultural change – have created a model of the kind represented thus:

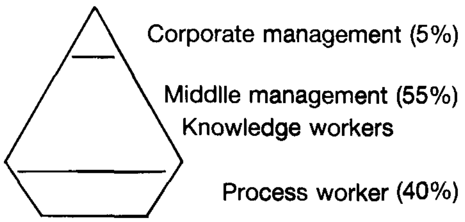


Fig. 7 Variation on traditional model

The tree structure in the middle management layer has superimposed upon it a diamond structure of practical working relationships to manage processes which are no longer compartmentalized into the same hierarchical structure as the Tree. The traditional line and staff roles are merging into the Knowledge Worker role.

Thus the 'real' organization is a composite and can be modelled as:

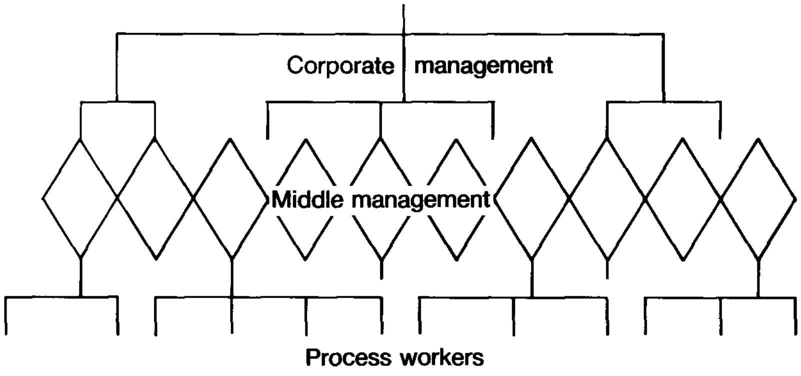


Fig. 8 Composite organisational model

This composite model is itself under further pressure, so to speak, as enterprises endeavour to squeeze out layers of middle management – to flatten the organization – and to reduce the total number of managers.

There are a number of key points about this model:

- Superimposed upon the 'Real' structure is the traditional 'Tree' – the hire/fire hierarchy – which all organizations still utilise to define management roles.
- The model describes an inherent instability which may be permanent or which may represent a transition state.
- Most importantly, it implies that fast and effective communication within the organization is crucial and, therefore, that data/information management and networking are key strategic issues.

Shifts in the nature of the establishment will also have an important bearing on IT support. As indicated above, this shift is twofold:

- Towards fewer process workers and more knowledge workers.
- Overall, towards fewer, better staff – relatively highly paid but providing better value for money.

All this focuses on the potential importance of IT as one of the means of ensuring that this critically important resource is used, and developed, effectively.

These concepts are summarised (with some additional material) in the following chart, presented in two sections:

Prime IT Strategy	Management Techniques	Application Objectives
SUPPORT	MIS is ZBB controlled. IT is only deployed where direct ROI is achievable. Reduction of the IT cost base is pursued.	Functional labour replacement is the prime objective.
FUNCTIONAL	The IT unit is an internal profit centre. Users are accountable for spend/ROI. IT contribution is functional unit efficiency.	Departmental Labour Replacement is the prime objective. Applications are bounded by organization and duplication is likely to occur.
BACKBONE	IS objectives/strategies and resource allocation are corporately defined. Corporately defined functional and data models are established. IT contribution is the provision of the network and data management.	Applications not organizationally bounded but support complete functions; efficiency, effectiveness of individual departments is not the prime objective.
STRATEGIC	IT forms part of the corporate plan and there is corporate management of IT programmes. IT contribution is improved market share.	External Orientation of data is pronounced. Decision Support Systems are in evidence.

Fig. 9 Summary of IT support strategies (Part 1)

Prime IT Strategy	MIS Approach	User Awareness
SUPPORT	Monopoly	Reactive; superficial involvement with little understanding
FUNCTIONAL	Free Market	Driving force; user accountability for value added contribution
BACKBONE	Monopoly of IT resource	Participatory; joint accountability of MIS and User
STRATEGIC	Free market with leading edge policy	Central direction and control by corporate strategy group with MIS participation

Fig. 10 Summary of IT support stretegies (Part 2)

This view of management strategies reveals several different points:

- That particular combinations of management techniques will lead to the development of particular aspects of application functionality.
- That the use of a mixture of these management techniques will probably give rise to conflicting objectives for application development.

It is important to note that different management approaches may well exist within an enterprise at any given time. For example, an enterprise may as a policy use the 'Support' strategy while a subordinate unit is using the 'Strategic' approach.

The essential point is that these circumstances need to be identified, justified and managed appropriately otherwise a somewhat schizophrenic enterprise may result.

Each of the strategies discussed so far is defined primarily as strategies for managing IT. They do not define an application set per se and do not attempt to do so.

The importance of choosing a strategy is that it allows the prioritisation of specific aspects of application functionality. It is the requirements of the competitive strategy that defines the application areas that have most priority and the IT objective that defines the priority functionality within the application areas.

By establishing priorities of application and functionality within the total potential IS portfolio, specific application strategies can be derived together with appropriate management methodologies to achieve those strategies. The following sections provide some insight into how this may be achieved, but first a few words concerning planning below the corporate-level.

3.6 *Functional-level planning*

Planning within the functional units below the corporate body essentially follows the same rules and should yield a MOSC set which provides more layers of detail. The MOSC set at this level should patently be consistent with the corporate MOSCs with no disconnects. A hierarchy of MOSCs is thus established with, broadly, one man's strategy being the next man's objective.

4 Transforming business objectives to IT objectives

4.1 *Introduction*

We now move to the next 'cloud' in the planning process which deals with the transformation of the corporate and functional objectives and strategies into selected IT objectives and strategies and a preliminary view of application requirements.

A review of the pictorial overview will remind the reader of the inputs and outputs. Some of these have been discussed above and we will now comment on the rest.

4.2 External analysis

It has been frequently observed that inwardly focused enterprises are likely, in due time, to find themselves falling behind the competition. Anyone working in Detroit, for example, will confirm this axiom. The external analysis helps to avoid this danger by gathering intelligence on how competitive organizations are employing IT to hone their competitive edge. The analysis should not be confined to competitors, however, as much of value can also be learned from relevant successful companies operating in other fields.

Intelligence sources range from periodicals, specifically commissioned and/or multi-client surveys, seminars through to 'debriefing' recently hired staff who are in a position to provide relevant information.

The information obtained needs to be treated circumspectly, as a slavish copying of competitors may be as dangerous as ignoring them.

4.2 The applications taxonomy

The IT strategy will necessarily place heavy emphasis on the role of applications – indicating which applications will and (importantly) will not be available during the timeframe of the IT plan to support the objectives and strategies of the enterprise. This is not to say that applications software is the only technology to be employed but, rather, that is likely to predominate.

The applications will require, of course, hardware, system software, networks and support personnel. Nevertheless, the applications are first-order components and, ipso facto, should be determined first.

In formulating a view of the required applications, an Applications Taxonomy is helpful and should provide the following benefits:

- It will help to avoid creating physical applications which are a mix of the logical types.
- It will help overcome the transaction processing mind-set that many data processing organizations tended to develop during the 1970's. This results in a tendency to still produce 'dumb' systems when 'smart' systems are needed to give real user benefits.
- It helps to decide on 'sourcing' – should the application be developed by the MIS development shop, by users via the Information Centres or should it be acquired/developed from/by an external supplier?
- As an aside, the taxonomy is also useful in highlighting the lack of adequate analysis techniques for many of the application types. It is the data system domain that has been generally well served while the information system domain has not.

A pictorial summary follows – a refinement of the application pyramid of Nolan et al – together with a definition of each application type. Unfortunately, given that the taxonomy is useful rather than overwhelmingly important, it is not possible to give a shorter representation without loss of clarity.

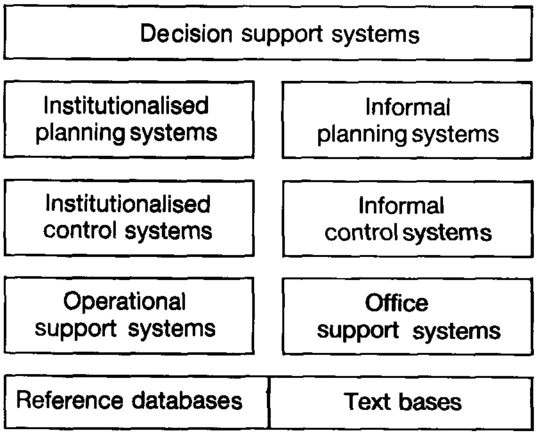

Decision support systems
Institutionalised planning systems
Informal planning systems
Institutionalised control systems
Informal control systems
Operational support systems
Office support systems
Reference databases
Text bases

Fig. 11 Summary of application types

Reference databases

These systems support the acquisition and recording of data resulting from/relating to internal administrative processes. They result in a 'database' containing data for use primarily by operational and control systems but which may also be referenced directly by people. Control of the Reference Databases is centrȁlised, although the data providers may be widely drawn from within the enterprise. Reference data may also be obtained from external sources.

They are typically transaction processing systems employing database management software. They are generally developed by data processing units rather than End-User Computing units such as Information Centres. These applications are essential to the functioning of the systems of the enterprise and, as such, they are of high business importance.

Text databases

These, as the name indicates, store information in the form of text for use by people rather than systems and may be accessed by direct reference and by keyword search. The contents are acquired both internally and externally. The establishment of these text bases may be handled by End-User Computing units.

Operational support systems

These systems support that base administrative processing of the enterprise which is directly associated with physical events – such as receipt of orders, delivery of goods, purchase of components, etc. They are almost exclusively of the transaction processing variety, processing data rather than information. They operate on a 'flow' basis, being available throughout the whole working day, processing varying volumes of transactions as required. They may also have batch processing elements, handling interfacing to other applications and the production of control information which is directly associated with the Operational Support System.

The systems are typically developed by data processing units utilising database management software and they are of high business importance.

The above does not adequately deal with specialised systems – device control systems – found in manufacturing and other environments. In the manufacturing environment, these microprocessor-based, real-time systems control the individual facilities of the automated manufacturing flow lines – such as computerised conveyors, carousels and robots. As such, they are a form of hybrid system in that, although clearly Operational Support Systems, they also take part in the physical effect itself. These systems are highly specialised and are normally acquired from specialist suppliers.

Institutionalised control systems

These applications are very closely associated with the Operational Support Systems from which they draw their data. They are information systems rather than data systems. They provide monitoring information – for communication to and between many organizational layers – by extracting, summarising and linking base data. The data are 'actuals' and the systems are run periodically – typically weekly or monthly. Normally produced by data processing units, the applications are essentially of the batch processing variety, although output may be available on-line.

They are key to the management process within large areas of the enterprise and are of high business significance. This and the regular nature of their operation gives rise to the term 'institutionalised' (c.f. Informal Control Systems below). These systems may take data from Institutionalised Planning Systems in order to report variance from plan.

Institutionalised planning systems

These deal with predictive information, providing important budgetting, forecasting and requirements information for communication to and between many organizational layers within the enterprise and, as such, are of high business importance. They may be produced by data processing units but, if so, will normally require the major involvement of key users to specify the algorithms which are at the heart of these applications. Alternatively, they may be developed by 'Information Analysts' within the user fraternity in close association with the MIS department as adherence to standards is of prime importance. The algorithms involved will be generally well understood.

Office support systems

These systems are hybrid in the sense that, although each element of the data/text processing they carry is normally classifiable into one or other of the definitions, the prime characteristic of an Office Support System is that it is generalised to allow particular 'offices' to perform combinations of activity. The term is used here in its narrow sense to cover word processing and voice/word/image/data transmission. These 'infrastructure' systems, judicially applied, may be of high importance to the efficiency and effectiveness of the enterprise. Their introduction is typically supported by End-User Computing units.

Informal control systems

These have some of the same basic characteristics as the institutionalised variety but serve the control requirements of individuals, teams, or small departments and do not directly provide data outside strict organizational boundaries. They are developed by users in association with End-User Computing units – typically without the strict adherence to standards essential to the institutionalised variety – and may well be personal computer based. They have low business importance and the failure of one of these systems should be no more than an irritant. They may well be transient in nature, dying out if, for example, the original creator moves to another department. Successful examples may increase in scope and importance and become candidates for institutionalisation. At this time, the lack of adherence to standards may become an issue and this, and increased volumes, may demand a re-write.

Informal planning systems

Again these have the same base characteristics as the Institutionalised variety but service the planning requirements of individuals, teams or small departments and do not directly provide data outside strict organisational boundaries. End-User Computing supported development is again typical and they have low business importance relative to the institutionalised form. Successful systems may also become institutionalised over time.

Decision support systems

This is a relatively new class of application, of high potential importance but currently shrouded in a degree of hype. In this taxonomy, a system which delivers processed data to a manager is not a DSS. The title is precise enough in that these systems support decision making processes. They tend to be based on models which provide the decision makers with a 'what if?' capability to illuminate the likely outcome of a particular decision within a set relevant to the decision domain. This assists the decision maker but still relies on his/her experience and judgement.

To be successful, Decision Support Systems must be developed by the user (normally in association with End-User Computing units) and may well start life as prototypes. If a system is successful and the underlying algorithms become well understood, it may generalise to become an Institutionalised Planning System. They may well utilise external data extensively and include 'soft' data (for example, City opinions, ideas, predictions).

Executive Support Systems are a variant of the application type, broadly having the characteristics of DSS but the users are top level management. A distinguishing feature is that an 'Information Analyst' will frequently develop the system in close collaboration with the executive, this giving rise to the term 'chauffeured systems'. However, it remains essential that the executive has the necessary vision and enthusiasm to act as the catalyst if these systems are to be successful.

The Application Taxonomy completes the consideration of inputs to the Transformation process and we now turn to the two complementary outputs.

4.3 Application objectives

At this point in the planning process, we are able to take an initial view of the application requirements. While we have used the term 'Application Objectives' on the pictorial view for reasons of simplicity, the real term is 'Application MOSC'. We believe that the 'MOSC' approach – Mission, Objectives, Strategies and CSFs – is as essential to the broad definition of an application as it is to the (partial) definition of an enterprise.

The use of the Application MOSC notion is an important aid in concisely defining the essence of the requirement and in ensuring that the subsequent development remains loyal to the business need.

Each Application MOSC will need to be supported by additional material, such as a 'Functional Model' and an 'Entity (data) Model' (or a delineated sub-schema of an existing Entity Model). As these components are fairly conventional in nature, they will not be elaborated here.

4.4 Selected IT support strategies

The Transformation process is also concerned with selecting the IT Support Strategy (or Strategies) from the portfolio presented above. These need to be appropriate to the corporate values and the requirements of the individual functions.

The development of the Application MOSCs referred to above will be 'filtered' by the IT Support Strategies, thus removing apparent application needs which are in conflict with the corporate and functional values.

The IT Support Strategies will also provide an early view of priorities and, hence, sequence of application development.

4.5 *The enterprise model*

Finally, in this section, the more stoical planner may wish to develop an 'Enterprise Model'. The elements of model are outlined below but the planner will need to trade-off its benefits against the increase planning time required. If taken to the extreme, the planner may thereby fail to complete one planning cycle before the next – with its inevitable changes – is upon him/her! Against this, the Model will prove a useful educational aid for MIS staff.

The Enterprise Model provides, in essence, a description of the enterprise. It should not only represent the current state of the company but also the anticipated future changes looking over the planning horizon (typically three to five years).

The main components of the model are listed – with brief explanation – as follows:

- The MOSC set – As defined before.
- Organization Chart – The organizational view of the enterprise covering (say) the top five levels, including numbers of staff by role.
- Products and Services – A brief description of the products and services which are traded by the enterprise (with approximate volumes), together with an indication of future products/services (and volumes).
- Customer Base – An outline of the type, number and distribution of customers.
- Supplier Base – An outline of the type, number and anticipated distribution of customers.
- Channels of Distribution – A summary of the means by which products and services are sold and distributed to intermediate and end-customers.
- Functional Model – A top level view of primary functions and data/information flows between functions.
- Entity Model – A 'global' data model defining the primary entities and the relationships between them.
- Information Model – Equivalent to the Entity Model but dealing with primary information rather than primary data.

The Prime IT Strategies should be born in mind when developing the Enterprise Model, as they may be used to focus on the more important aspects of the business. For example, it may be unnecessary to chart in detail sections of the business which will be operated on a 'Support' basis.

We now move to the next 'cloud' to consider how these outputs may be embellished by second-order elements to provide a more complete view of the systems requirements.

5 Transforming the Application MOSCs into the IT Architecture

5.1 Introduction

The Application MOSC set provides the initial perception of required applications and we now need to transform this into a proposed 'IT Architecture'.

The first part of the process is to prepare a Preliminary IT Architecture by adding the second-order components. This is a creative process which will also employ information gathered in the 'External Analysis' and from a review of the 'Technical Opportunities'. The former has been covered earlier so we will now deal with the latter and with the format of the output.

5.2 Technology opportunities

The technology available at any given time will provide both opportunities and constraints and it is necessary to create and maintain an overview of available products (and methods, tools and services). It is necessary to have some view of future change, althouth this clearly needs to be approached with caution.

A technology description will help to ensure that the IT Architecture, targeted for a specific future period, can actually be turned into reality and that the available technology is fully exploited within the boundaries of the chosen management approach.

It is necessary to have a future view of the likely opportunities – the trick being to distinguish between that which one would like to have against that which one will probably get. Vendors' Statements of Direction may be helpful here, although these should be weighted by a consideration of past delivery performance. Other information gathering means include literature surveys and the better seminars.

It is proposed that the key elements of the technology are described in the form of a set of 'Product Descriptions'. In essence, each document should contain a brief product description, a pithy SWOT analysis (Strengths, Weaknesses, Opportunities and Threats), the expected availability of the technology (if not already available) and a statement of any product dependencies.

5.3 The preliminary IT architecture

In essence, this specifies the initial view of the complete applications systems base, with the supporting hardware, networks, etc. – which should exist at the end of the planning horizon.

Planners with a flair for the graphic arts and with access to a graphics workstation come into their own at this point, as the Preliminary IT

Architecture is best handled in pictorial form – backed-up by the Application MOSCs.

Applications are presented as black boxes, with the major interfaces and data/information flows shown, and each application should be supported by an outline data model and a functional model backed-up by brief functional descriptions.

A further chart should be established indicating the supporting hardware required, the operating systems employed, the networks and networking devices and the supporting personnel (to operate and maintain the systems).

In addition, it will prove beneficial to indicate major new applications which have been identified but which, at this preliminary stage, are unlikely to be constructed within the planning horizon. This can assist in managing users' expectations, particularly those who have not been directly involved in the planning process.

Finally, the applications within the Architecture should be given a relative priority rating, as indicated by the Selected IT Support Strategies.

This preliminary view must now be further filtered through the 'Systems Baseline' and the 'Business Case' to produce the 'Proposed IT Architecture' and the 'Investment Plan'.

5.4 The systems baseline

In essence, this is the current IT Architecture, as output from the previous planning cycle. Its consideration is deliberately delayed in the planning process to help prevent mind-sets from constraining the initial systems thinking.

The Baseline will influence the Investment Plan, as much of the required architecture may be more or less present. It may also be a constraining influence, however, particularly where a significant change of business direction has been deemed desirable.

5.5 The business case

The selection of a target IT Architecture involves the mixing of a number of ingredients in a process which is itself a mixture of science, craft and art. The Business Case holds the reasons why the application choices were made.

Firstly, the rationale behind the thought processes involved in the decision should be documented. This should provide a link back to the relevant corporate/functional objectives and strategies and to the IT Support Strategies. For example, if the enterprise has an objective which will cause it to adopt a niche marketing strategy, then a marketing database would be a logical application requirement.

Secondly, a benefit analysis should be provided. However, the quantification of costs and benefits remains a considerable problem. Costs tend to be more immediately tangible but development expenditures are frequently underestimated and implementation/operating/maintenance costs frequently ignored. Benefits, on the other hand, tend to ignore external (to the application) variables which may influence the achievement of the declared benefit.

One thing is becoming clear: it is becoming increasingly difficult, with the general spread of computing, to identify an application where the 'hard' benefits alone provide an attractive justification. In most cases, return on investment (ROI) analysis involving an algorithmic consideration of costs and hard benefits may well result in a low or negative rate of return. This often leads to a massaging of benefits or costs (or both) so that the 'desired' answer is achieved.

More often than not, 'indirect' and 'soft' benefits must also be considered. The most appropriate approach is to present each element of the cost-benefit equation discretely and establish a 'decision forum' in which the decision may be taken by a suitable management representative of the unit providing the funding.

The elements of the analysis are represented as follows:

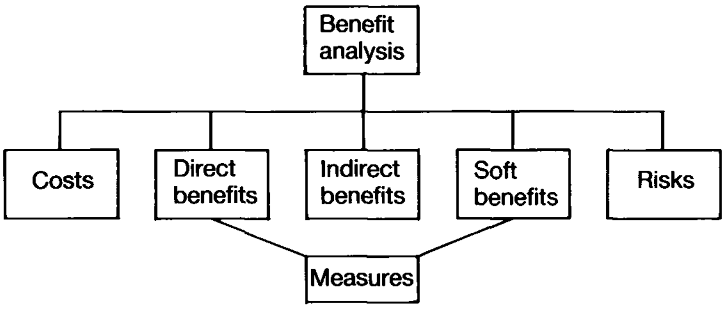


Fig. 12 Benefit analysis elements

The following comments expand on the picture:

- Costs – As indicated above, costs should include the full costs of the application, covering development, implementation and all operating costs.
- Direct Benefits – This covers quantified benefits where headcount-reduction and/or capital-employed-reduction is involved.

- Indirect Benefits – Here, avoidance of headcount-increase and/or capital-employed-increase is the primary argument.
- Soft Benefits – These are benefits for which it is not possible to provide quantification which is auditable. They are therefore necessarily expressed qualitatively. The declared values of the enterprise – such as its quality ethic – are of significance here, as they will tend to provide weight to soft arguments.
- Risks – A statement of the risks involved in the proposed project which may adversely affect the benefit analysis.
- Measures – For Direct and Indirect Benefits, the proposed measures to be employed are specified. In certain cases, there may be a disconnect between 'end' and 'intermediate' benefits. For example, a new sales order application may be justified on the basis that it will increase sales revenues (and hence profits) by freeing sales staff time through reduced administration. Simply measuring sales revenue before and after implementation is unsatisfactory because too many uncontrolled variables will be operating. An indirect measure such as sales administration per order-line may be deployed but note that, if more time is made available for selling, the opportunity may not be fully exploited.

The relative importance of the elements of the analysis is heavily influenced by the Prime IT Strategies. To illustrate this point, a 'Support' approach is only concerned with Direct benefits while a 'backbone' approach puts weight on Indirect and Soft benefits.

The Business Case provides a filter on the preliminary view and modifies the Proposed IT Architecture accordingly. The process may well be iterative as alternative means of achieving the application requirement by more financially attractive means are explored.

5.6 The proposed IT architecture

This is simply the filtered version of the Preliminary Architecture, expressed in the pictorial form outlined above and including the Application MOSCs, modified as necessary during the filtering. In particular, the Strategy element of each Application MOSC will be enriched as the means by which the application functionality will be delivered will by now be clear.

5.7 The investment plan

This is an elementary transformation of the Business Case and the Architecture into the 'house-style' for the enterprise.

6 Concluding remarks

We have now completed the main body of the planning approach and need only to cover a few remaining items. As these will be more than familiar to most readers — and the detail will, in any event, be dictated by company style — we need only cover them as concluding remarks.

Within the Corporate Review process, the IT Investment Plan will be reviewed with all other investment plans and the necessary trades-off made. This may well result in iteration and further re-work to the outputs described above.

The IT Architecture effectively defines what is required and the Business Case indicates why. Migration planning completes the picture by providing the detail on when, how and who.

The timescale required to develop a strategic plan will clearly be a function of scope and depth. However, a working rule is emerging which indicates that much more than 4 months is generally unacceptable. This appears to be due to a combination of 'attention span' and the fact that the degree of business change which may occur over long timeframes results in the planning process becoming fundamentally unstable.

The time to update an existing plan should clearly be rather less and should be undertaken with a frequency which reflects the volatility of the industry sector. Quarterly or half-yearly is likely to prove the most appropriate frequency.

A critical success factor is that the ownership of the planning team must be consistent with the approach identified during the Initial Orientation. The MIS unit will probably own and staff the team for 'Support' environments while it is essential that MIS does not own the team for the other approaches. It may also be necessary to enlist the support of a respected consultancy firm, this tactic being of particular value if 'political' factors are operating.

This paper is not concerned with the execution of the plan but we would propose strongly that the Application MOSCs are a crucial aid to maintaining the right bearing and speed in the stormy waters which will be encountered in even the most controlled and stable enterprises.

How often have the original business objectives been lost during application development? Most readers will surely answer 'All too often!'. And some may add ... 'If they were ever really understood at all!'

Preparing the organisation for IPSE

P.W. Veasey and S.J. Pollard
ICL Group Information Services, Putney, London

Abstract

In May 1985 the Group Information Services Division of ICL started an investigation into Integrated Project Support Environments (IPSEs). Based on that work, this paper proposes some common ground for organisations who are preparing for the extensive automation of software development. It offers an integrated view of technical and management issues in an attempt to provide a common framework for the definition of support environments. The authors hope that this may facilitate orderly evolution of methodologies.

1 Introduction

Integrated Project Support Environments promise software developers high quality and productivity through the integrated automation of software projects. Some IPSE products already exist, others of greater power and scope are promised. Pushing aside the hype, what is the position of the potential user faced with the total problem, and uneasy about the significance of the gaps in existing and projected IPSE products? "Does this product cover everything? Will it integrate with other working practices? Will it only work optimally with specific methodologies for design, planning etc., or with specific organisational structures? Shall I be locked in?"

In preparing their organisations to get the most out of IPSE, software development managers must recognise that they will finally be expected to take some of their own medicine – extensive automation. Potential IPSE users will discover that they are themselves no different from the target users of their own previous creations. They will need a much deeper understanding of the process to be automated and a much more precise vocabulary than was necessary under a "manual" regime. Unfortunately, the rate of change in computing has militated against the recognition and wide acceptance of such vocabulary and concepts as would give us all a common understanding of what is going on in our "cottage industry". This paper deals with this aspect of preparing for IPSE. It offers a view of the environment which we would regard as sufficient for the introduction of the rigour necessary for extensive automation, yet restricts itself to the issues which would be common to most software development environments. It should be of particular interest because of the extent to which Management issues have been integrated with

Technical ones. Within an organization, wide understanding and acceptance of such a view would seem to be a necessary precursor for the successful introduction of IPSE.

2 Some history and context

The view presented here is derived from work carried out on a project aimed at preparing a large "Information Services Shop" for IPSE. The IS Shop in question is the Group Information Services Division (GIS) of ICL which exists to provide the company with the systems for running its business. The project was carried out by the Strategy and Technology Unit of GIS. It set out initially to gain an understanding of IPSE and to formulate a strategy for its introduction. It looked at existing IPSE products such ISTAR, MAESTRO and BIS IPSE, and the work of the Alvey Directorate towards an advanced IPSE. The latter, in particular, made it clear how much was at stake. The emphasis shifted, from an interest in early deployment of what was currently available, to an attempt to strategise our methodology development with a view to its eventual convergence with the future IPSE products which could enable the really dramatic improvements. This required some guesswork as to just what kind of environment, particularly from a management point of view, these future IPSE products would fit into. This and the prospect of trying to manage the evolution of our development methodology brought us quickly to the realisation that we had to have a much deeper understanding of the project process for software development. The need to closely integrate the Management issues with the Technical made our search for an appropriate model very difficult. The methodologies we studied such as LSDM (SSADM, ISDM) Jame's Martin's Information Engineering and CACI did not go far enough in explaining how planning, quality and project control integrated with analysis, design and coding. IPSE developers, who might have been expected to have already given this much thought, were not forthcoming. Whether this was because of secrecy or because their product orientation led them to concentrate on the technical side we were not able to establish. The project, therefore, spent considerable time creating its own model of the software project and its support environment (Ref. 1).

This model was then used in defining a specific environment suitable for that part of GIS's mission concerned with producing large commercial systems. This was done by writing a procedural document, SENSE (acronym for Support Environemt for New Systems and Enhancements) covering the whole enviroment, which gave substance to that definition. The document conformed to the model but defined the environment fully by specifying the particular methodologies and policies chosen. This document has now been passed over to the Development Unit where an implementation project has been initiated. This implementation project is now proceeding with a good head of steam and with consultative support from the Strategy and Technology Unit.

Having completed the writing of the first version of SENSE, it is now possible to see what elements of our earlier modelling were essential in keeping methodology development under control. These elements, modified by the experience of writing SENSE, have been extracted to produce COMMON SENSE, the name given to our "view" of the environment.

This paper then proposes some common ground for all those involved in automating software development. The view it presents of the project support environment corresponds to a database schema in that it attempts to cover all aspects of the environment. More traditional approaches could be seen as "subschemas", corresponding to the view of the environment from a particular role. For example, a Project Manager's subschema would cover most of the issues dealt with in a more traditional approach to Project Management. Human issues, however, such as negotiation skills and man-management would be noticeably absent. This is not because they are considered unimportant but because such issues "decouple" in a satisfactory way from the mass of interacting issues which an IPSE is intended to automate. They are therefore not part of COMMON SENSE and so cannot be in any sub-schema of it. Other important issues which are absent for the same reason are the distribution and MMI aspects of IPSE, and money. Evaluation of resources in money terms is, after all, not conceptually difficult and finance issues decouple from those of environment.

COMMON SENSE should be of interest to any software development organisation that is striving to create an environment in which its projects can achieve high quality and productivity. We would claim that COMMON SENSE is universally applicable, although it relates most obviously to organisations which expect to be ongoing and are not set up purely for a single project. It represents a totally general model of the software development environment with the single constraint that it must enable high productivity/quality. One might argue about the names given to the concepts but the concepts themselves seem to us inescapable. Possible consequences of keeping to common ground might be either that there is very little substance to COMMON SENSE or that much of what it states will be motherhood. Our experience is that reactions to it vary widely in these respects. For us it is not an issue since we are convinced of its benefits.

3 Benefits to be expected

The value of establishing agreement on these underlying concepts and vocabulary is enormous:

- alternative methodologies/environents can be more easily compared.
- environments can change, improve, and adapt much more quickly without loss of control.
- large organisations (such as ICL) whose different divisions need different environments, can at least have some common language, which may facilitate sharing of resources/tools and consolidation of reports. It may

also be possible for a central department to guide the development of multiple environments towards some common future environment. This may be desirable where we are talking about advanced IPSE's which are likely to be extremely expensive.
- a piecemeal approach to automation of the environment becomes controllable. This is desirable since no organisation is likely to find all they want in any available product. Before COMMON SENSE we were uneasy about introducing any automation since we could not see clearly the scope of its impact.

4 Common sense

4.1 Method of presentation

The view that COMMON SENSE takes of the Project Support Environment is developed here through a series of "concept maps" which show how the various issues are related in the COMMON SENSE view. They represent a particular choice, for the purposes of this paper, from all possible partitions of the "total" map. This "total" map, which the reader may reconstruct by joining the individual maps together, was of great importance to us in choosing a structure for the SENSE document. The concept maps are given here so that interested readers may use them to conduct a similar exercise.

The technique is intended to maximise communication rather than rigour. The relationships indicated by the arrows are not all logically equivalent and can only be understood from the accompanying text. The "types' of the concepts are also very mixed; some are "activities", some are "requirements". Our work has included more rigorous, though partial, expositions of COMMON SENSE. These have satisfied us that enough rigour can be achieved to guide automation, albeit at a cost in immediate understandability.

In what follows, the various topic areas of COMMON SENSE are discussed in turn, with concept maps providing an informal introduction to the terminology surrounding each topic. The topics heading are:

Scope of COMMON SENSE and the role of Guidance.
Guidance Classification.
Control and Components.
Distinguishing WHAT from HOW.
Contracts.
Quality.
Standard Networks and Methodologies.
Iteration.
Re-Usability.

Terms which have already been included in a previous concept map are underlined when they reappear.

4.2 Scope of COMMON SENSE and the role of Guidance

As an introduction to concept maps, some explanation of Fig. 1 may be helpful. Most of the relationships used in the concept maps are those of "conceptual precedence", i.e. "You must understand what we mean by A, before you can understand what we mean by B". Occasionally, it indicates that an implementation of A is necessary before B can be (fully) implemented. With reference to Fig. 1 below, this tells us:

(i) Knowledge of our view of the Software Life Cycle, and our bias towards Development makes understandable our view of the Project Life Cycle and its terminology.
(ii) Before classifying Guidance we must understand what is meant by guidance.
(iii) The structure of a standards taxonomy will be influenced by:
- Our view of the Project Life Cycle.
- Our Guidance Classification.
- The requirements for automation.

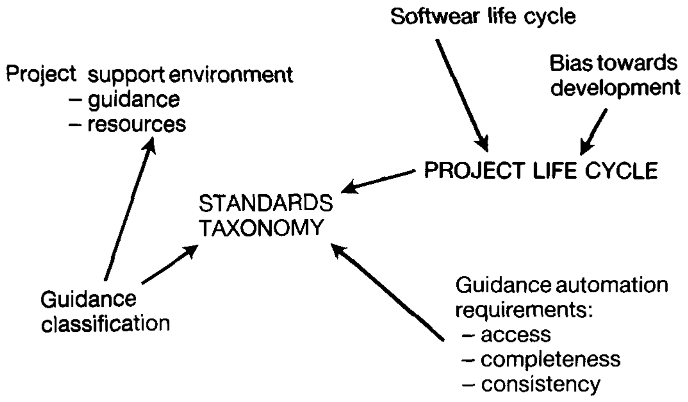


Fig. 1

The first step towards rigour must be to make clear what we are talking about. COMMON SENSE is about Software Developments Projects, but, interestingly, nearly all of COMMON SENSE would be valid for any project. We start with a very simple view of the Software Life Cycle as below.

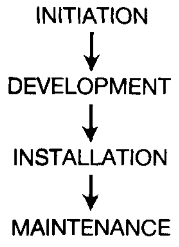


Fig. 2

Our bias is towards the "Development" phase which is seen as starting with a firm commitment to some Initial Specification and ending with the delivery of an installable product. Development then is the project on which we focus our attention resulting in a Project Life Cycle as below,

$$\begin{tikzcd}
\text{PROJECT INITIATION} \arrow[d] \\
\text{PROJECT PROCESS} \arrow[d] \\
\text{INSTALLATION}
\end{tikzcd}$$

Fig. 3

where Project Process in this case equates to Development, and Installation may occur any number of times as separate projects of a different type. At some stage it is intended that we should extend SENSE to cover the Installation and Maintenance phases. SENSE is already linked to a methodology for IT planning: this is described by Pollard and Crawford in this same issue (ref. 3) and takes us through the initiation phase to the point where SENSE takes over.

The requirements for supporting a software development project are seen as "Guidance" and "Resources". Guidance may be mandatory or advisory, e.g. respectively, a standard MMI and a recommended training course on data modelling. Resources are either physical things such as offices and computers, or software such as operating systems and electronic mail systems. Some things in the support environment may share both natures, for instance, syntax-directed editors. One of the most important features of an IPSE is seen as the "automation of Guidance". There will always be limits to this but, to stand any chance at all, Guidance must be held as an accessible, organised set of standards according to a standards taxonomy. The classification of Guidance must be seen to be comprehensive and the disciplines of the taxonomy must ensure consistency.

The standards taxonomy then structures all the Guidance. In a very advanced IPSE, most of the Guidance would be available on-line for enquiry and as a knowledge base channelling the user to adopt correct practices. To grow to such an IPSE it should be obvious that developing a standards taxonomy, and writing the standards, is an essential step. It is equivalent to achieving complete and accurate manual stock records before computerising Stock Control.

4.3 Guidance classification

Organisations devoted to developing software are seen as having two main functions. Firstly the creation and maintenance of an environment which can support projects. Secondly the initiation and execution of the projects

Guidance Classification

Environment Maintenance
- Project Database
- Organisation
- Base Environment
- Quality System

Project Activity
- Production
- Planning
- Control

Fig. 4

themselves. It is also recognised that some large projects will split into multiple projects. They do not require any extension to COMMON SENSE but, at a lower level, additional planning and control methodology must be put in place.

Project Activity is activity which contributes directly to a specific project or projects; project workers will, in addition, have to follow the rules of Environment Maintenance in order to preserve it in good order for the use of other projects. It is also likely that in an organisation devoted to software development there will be some people dedicated purely to maintenance and enhancement of the environment. As indicated above, Guidance on Environment Maintenance must cover five issues:-

Project Database	Data Dictionary, plans etc.
Organisation:	recommendations on roles and organisations for projects and for specialist groups maintaining the environment, such as data administration.
Base Environment:	covers work environment e.g. office space, and computer environment e.g. computers, re-usable component database, configuration management software.
Quality System	methodology which assures the attainment of required quality levels.

This classification of environment maintenance appears to provide good cover with little or no overlap depending on precisely how the terms are defined. We would freely accept, however, that this may not be the only satisfactory classification and we do not claim to have followed rigorously every possible interaction of the elements. We are deliberately resisting the temptation to claim that the issues are disjoint or orthogonal. A number of people have proposed classifications which show the environment as a multi-dimensional matrix, but we have found that such representations cannot support rigorous work and that the comforting prospect of simplicity that they promise is illusory.

We have achieved greater rigour in our classification of Project Activity where, after lengthy analysis of the Project Process, we could precisely classify any project activity. in fact we would claim that the Project Activity classes are disjoint. A full exposition of this analysis is not appropriate here but it is reflected in the classification of components in the next subsection.

4.4 Control and Components

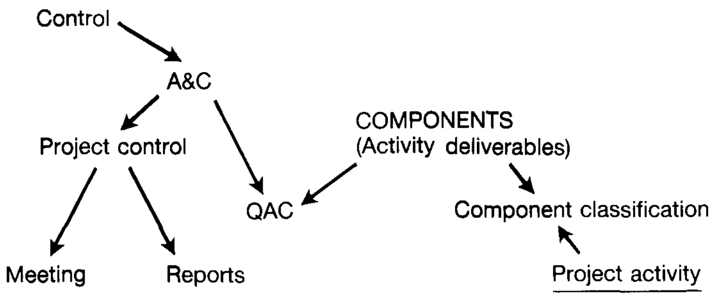


Fig. 5

In the classification of Project Activity, Control gets broken further into Project Control and Quality Assessment & Control (QAC). Both are examples of a general process of Assessment and Control (A&C) as below.

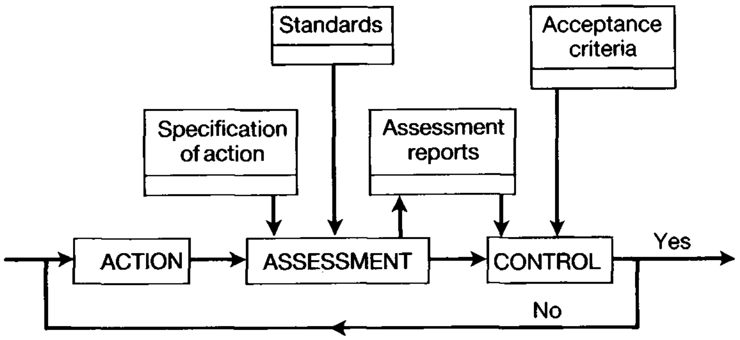


Fig. 6

QAC applies this process to components, the deliverables of project activities. Project Control, on the other hand, applies it to the whole project, so that in this case "Standards' and "Acceptance Criteria" correspond roughly to "Plan", while "Assessment Report" corresponds to "Progress Report". In both cases the repeated action may be modified by the control decision.

The formal aspects of Project Control can be summarised by a Meetings/Reports table, as in the simplified example opposite, which defines the frequency and distribution of reports and the frequency and attendees of meetings.

REPORT NAME	Freqy	Project Manager	Unit Manager	Project Board
Milestones	1 Month	——— ———	——— ——→	——→
Costs-to-date actual v. budget	4 Weekly	———	——→	
MEETING NAME				
Monthly progress	1 Month	✓	✓	
End-of-stage	N/A	✓	✓	✓

Fig. 7

A statement of the contents of each report, the agenda of each meeting, and which reports are required for which meetings, completes the definition of a chosen Project Control methodology. Once again the reader is reminded that we believe such issues as man-management can be de-coupled from COMMON SENSE.

The term "component" is crucial in COMMON SENSE. Components are the deliverables of project activities. The project activity classification is combined with the idea of components to give a component classification. An important class of objects emerges, the 'product components' which along with their metadata are the usual contents of Data Dictionaries. These product components contain information which gets built into the final deliverables of the project. "Production" in the project activity classification is then seen to be, more strictly, "Production of Product Components", so that production of a plan would be excluded.

4.5 Distinguishing "WHAT" from "HOW"

Having earlier castigated those who offer simplistic analyses of the software development environment, we now risk similar criticism ourselves. This is necessary because a completely unstructured standards taxonomy would be unusable. The WHAT/HOW distinction which is introduced here, and used in further structuring the taxonomy, does at least have some timelessness about it. We are trying to avoid using structures which may be short-lived, such as a particular choice of project organisation. If standards are organised according to who will use them, an organisation may be tempted to hang on to project roles which are no longer the best choice given the advances in technology.

The standards taxonomy is already 2-dimensional in that we can group

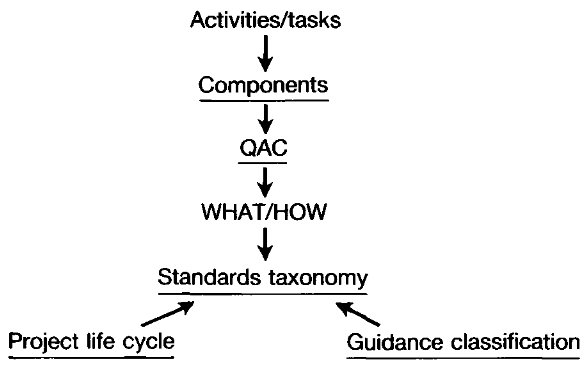
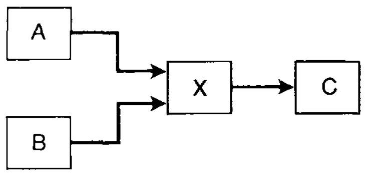


Fig. 8

standards according to their applicability within the life cycle and their position in the support environment (guidance class). To make it more usable however it is likely that another dimension is needed which will divide standards vertically between those that say *what* to do and those that say *how* to do it. A little thought will soon make such a distinction appear illusory since describing how to do something will consist mostly of saying what to do at a lower level of detail. Where then should we draw the line? In fact the distinction can be re-established in a satisfactory way. The key is in the level of "activities" described by the planning process. In COMMON SENSE it is important to understand that "activity" means an action which appears on the project's activity network and all of whose precedence relations are defined as for activity X below:



X cannot start before A finishes
X cannot start before B finishes
C cannot start before X finishes

Fig. 9

The "tasks" which make up an "activity" may be defined by the plan but are treated as a list, with nothing to constrain the order in which they are carried out. It is a principle in COMMON SENSE that the planned duration of any activity, at the moment it is about to be started, must be less than some limit set by policy (for GIS currently 10 days). This limit is set so that there is only an acceptable risk in always waiting till the end of an activity before imposing any formal control. This means that Assessment and Control need only be applied to components (the deliverables of activities) and not to the activities which produce them. "WHAT to do" then refers to the names of activities at this level. "HOW" is the detail of their execution. This gives us our WHAT/HOW distinction and additionally simplifies the formalisation of QAC.

4.6 Contracts

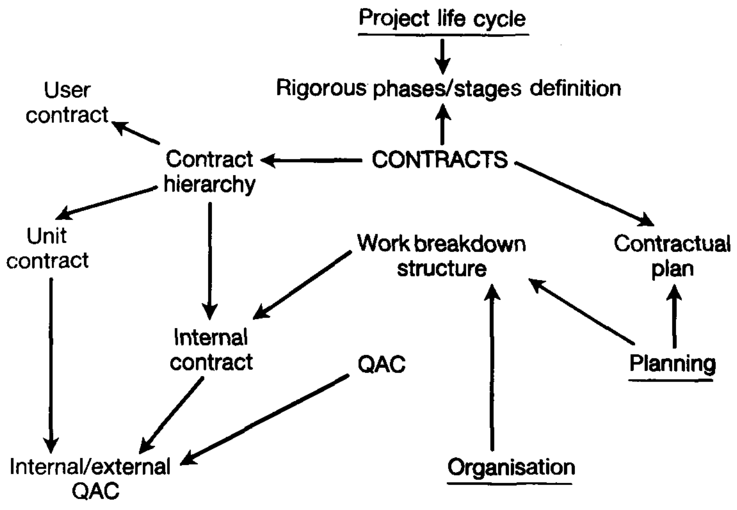


Fig. 10

The use of "contracts" is crucial in instilling a natural rigour and discipline into the project process. The "contracts" concept as used within GIS has grown out of the work on IPSE and is a story in its own right (see Ref 2). Suffice it to say here that it is a more rigorous form of what might previously have been "Terms of Reference", and has the following sections.

Client/Contractor
Purpose
Specification
Standards
Acceptance Criteria
Schedule
External Dependence/Assumptions
Resources
Reporting requirements

Note that statement of External Dependencies is the key to multi-project management and that the use of the word "contract" and the insistence on signatures is essential if the natural disciplines associated with contracts are to be achieved.

To achieve an environment in which rigour is possible, contracts must be used to provide clearly defined boundaries to the phases of the life-cycle and to the stages within those phases. This does not necessarily mean that there will be no overlap, but that it should always be clear which contract authorises a particular activity.

The extent to which overlap is allowed should be a function of the savings it can yield and its controllability in a particular environment. Our current implementation of this in SENSE demands that there should be no overlap of stage contracts. This is partly a question of control and partly reflects a belief that 4GL's make it possible for a small team to carry out all the stages of development. One should not, therefore, find a coding team waiting for a design team or other similar situations which are used to justify the breakdown of stage discipline.

The schedule of a project contract under COMMON SENSE is called a Contractual Plan and must conform to the standards for plans.

It is useful to create "subcontracts" within a project and some form of contract hierarchy will occur naturally. A minimal arrangement is likely to be:

A User Contract between the development unit and whoever represents the end-user.
A Unit Contract between the Unit Manager and the appointed Project Manager.
Internal Contracts controlling delegation of the project work by the Project Manager.

Where "development unit" equals "software house", the User Contract is a Sales Contract and has legal status. Failure to establish something similarly well defined and agreed is at the root of many of the problems encountered by internal IS Shops in their dealings with their users in the rest of the organisation. In either case we believe that "contract" discipline should extend deep into the project. The specification sections of Internal Contracts may correspond to work packages of a work breakdown structure, which defines the allocation of work in a project by tabulating the product decomposition against the project organisation.

Contracts help to formalise the project boundaries and clarify the QAC processes. Internal QACs are carried out within the Project Manager's auspices. An external QAC is applied at the Unit Contract level and a pass may release the component to the world external to the project, the world of the user. This concept of external/internal QAC has been judged useful and necessary for establishing control within GIS.

4.7 Quality

COMMON SENSE assumes that a "Quality System" is in place in the Support Environment. It carries out Quality Audit which checks, not only that quality standards and procedures are appropriate, but that they are also being adhered to across the whole environment. Although Quality Audit uses sampling and may therefore check the QACs which have been carried out on particular components from particular projects, its view is neither the component nor the project but the environment.

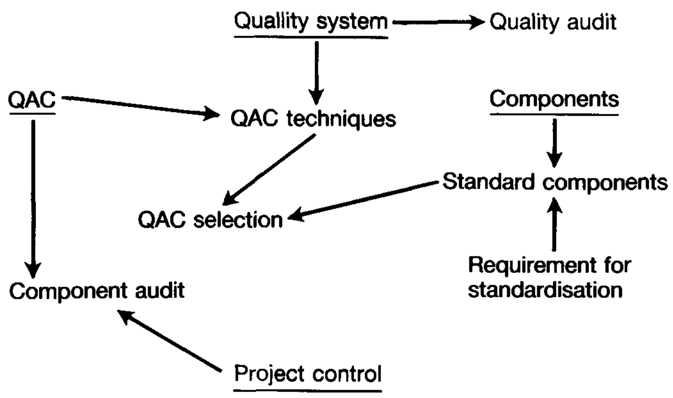


Fig. 11

Some of the most visible quality-related work on a project is not QAC but checking that QAC has been completed correctly. This may be done as a prelude to the client "signing-off" as acceptable a particular project deliverable. Here the focus is very much on particular components in particular projects so that we have called it "Component Audit" and classified it as a project control activity.

Broadly speaking, methodology is the wisdom gleaned from previous projects as to what activities should be carried out, in what order, to produce what components. Naturally each new project must be viewed on its own merits, and mechanisms must be in place to allow the evolution of standards to meet new situations. At any time, however, most or all of the components must be standardised to take advantage of methodology. The Quality System can then supply standard QAC Techniques which may be applied to standard components, and specify for each component which of these techniques should be selected. (Note that this may be more than one). For instance, Fagan Inspection Technique:

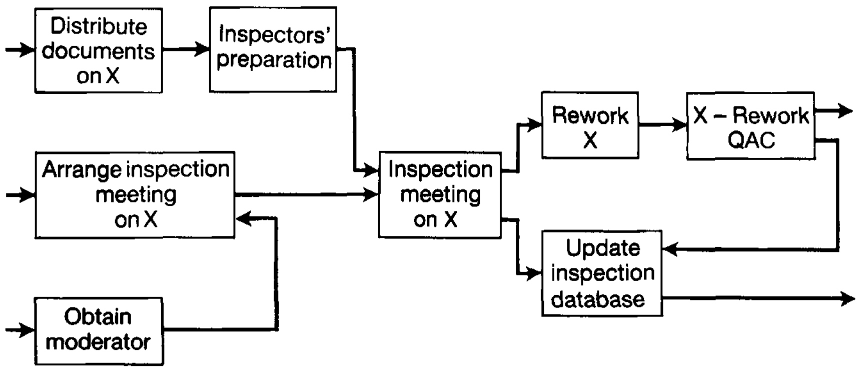


Fig. 12

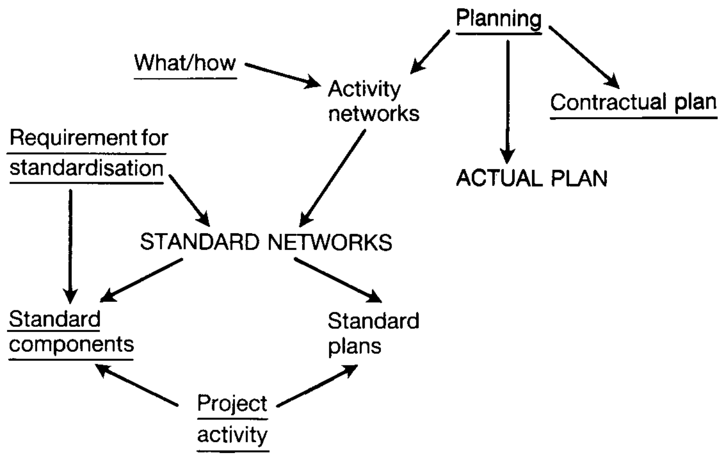


Fig. 13

In non-automated planning environments, one can usually find a formally recorded plan corresponding closely to the COMMON SENSE idea of a Contractual Plan. Parallelling this is something which may exist only in the project manager's head and which is the day by day view of how the Contractual Plan will actually be achieved. With adequate computerised planning tools, this "Actual Plan" can be more conveniently recorded, updated and manipulated on the computer. Such tools are now becoming available and it is assumed by COMMON SENSE that they are being used since better, more responsive planning is expected to be a major contributor to future productivity improvements. There should therefore be, in addition to the Contractual Plan, an Actual plan which is constantly changing in response to progress, etc. It will also be found convenient to store data on "actuals", such as days-worked and resources-used, against the activities on the Actual Plan. The many ways in which these two plans interact is explained in our planning methodology. For instance, if a point is reached when no possible change to the Actual Plan will result in achievement of the Contractual Plan, a renegotiation of the contract is unavoidable.

There are a number of ways of formally recording plans e.g. bar charts and activity networks. More advanced forms of representation may evolve in the future but for the moment, it is assumed in COMMON SENSE that we use activity networks since they represent minimal information for the automation of scheduling.

These activity networks, and the requirement for standardisation, lead naturally to the idea of standard networks. These effectively capture pieces of methodology. Standard components are WHAT to make. Standard networks show WHEN they should be made by ordering the activities which produce them. We have already seen a piece of QAC methodology (Fagan Inspection) captured in this way.

A particularly useful special case of a standard network is the Standard Plan. This should be referred to at the beginning of each stage and covers the whole Project Process from start to finish. It would normally be set at an intermediate level of detail and should identify all the standard components as deliverables of its activities. Its value is not only in planning but also as the central vehicle for defining production methodology Each stage of the Project Process is covered in the Standard Plan to the same level of detail, so that early in the project, its later stages would appear in the Actual Plan in summary form. It may also be the case that a development unit requires different standard plans for different kinds of development such as:

* Transaction Processing/Database Applications
* Expert Systems
* Package Modification
* Decision Support Systems

Whenever the unit undertakes a different type of development, consideration should be given as to whether the plans can be re-used and standards evolved by stating them as a Standard Plan for this project type. Automation of the deployment of standard networks can ease the burden of detail in rigorous planning.

Few people would dispute that issues such as "Planning" should be part of the kind of view COMMON SENSE claims to be. Some of the issues presented, on the other hand, appear to be part of our particular implementation of it. Our claim is that such issues as "standard networks" and "contracts" are essentially unavoidable in a high quality environment although they are frequently not addressed, to the detriment of the environments concerned.

4.9 Iteration

One of our early preoccupations in the work on IPSE was to understand the impact of prototyping which some were championing as an alternative to structured development. Our conclusion was, in fact, that the two were not incompatible. We built an iterative model of project activity[1] and tried to understand prototyping through clear ideas on the more basic concept of iteration. What became clear was that nearly all human activity, let alone project activity, can be thought of as iterative, if one recognises the rapid feedback loops which may be involved. It becomes largely a question of degree. For example, one would expect to view walking as a non-iterative activity but it could not be done without feedback loops to control body movement. Nevertheless, no-one would expect this level of iteration to be visible in a project. At the other extreme, work on the "2nd Release" of a

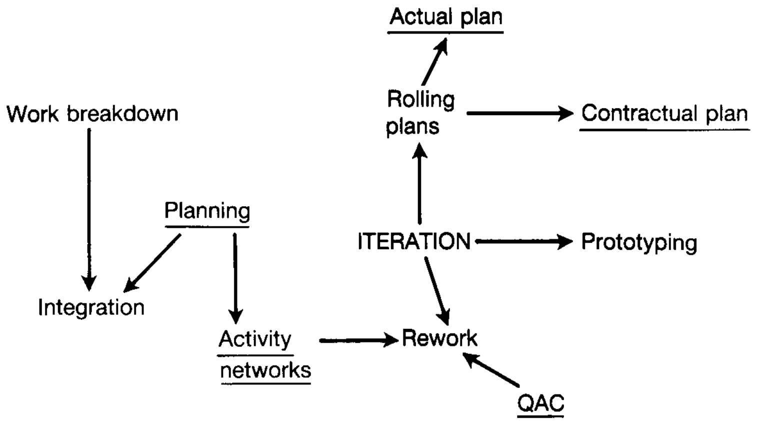


Fig. 14

major application may well be a project in its own right and yet iterate much of the original work as well as adding new functionality.

We concluded that there was a general problem about iteration at the "activity" level in projects, and a particular issue with regard to the activity of planning. The general problem arises from the fact that any practical activity modelling technique, which can be used for automatic calculation of project schedules, is unable to cope with looping – that is, "the repetition of activities an unknown number of times". The looping seems to be of three types.

Best understood is "Rework" which occurs as the result of formal quality checking. We have made it a principle that, in planning QAC activities, we assume that the single QAC activity shown in the network includes any rework and repeated QACs. The value of this is that it discourages people from using up the whole of the total time allowed for production before submitting the deliverable for QAC, then finding that more time is needed for rework. In a few cases, rework can be expected to take a long time, in which case separate rework activities should be planned.

Less well understood is work which is repeated as part of integration. The diagram below demonstrates a generic situation where a job is first broken down into constituent parts (Xi) which must then be reassembled/integrated.

Integration, in general, requires changes to the Xi, not just adding the "glue" to build X from its constituents.

Least well understood is the iterative action involved in prototyping. Rapid development methods can blur the distinctions between analysis, design and implementation. For some systems, elements of all three may occur in a single planned prototyping activity, with no intermediate QAC. From the fact that we are now talking about production methodology it will be seen

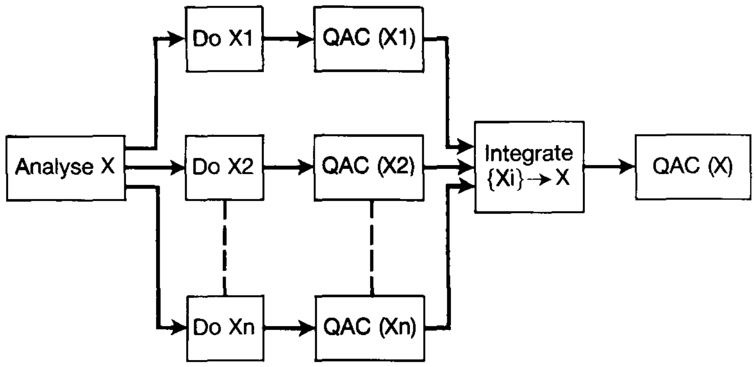


Fig. 15

that these problems occur at a level below COMMON SENSE. The SENSE document explains the way in which prototyping can be used without losing all the rigour of structured development. For our large systems environment it is seen essentially as an analysis method which is directed in the first place at achieving a correct Requirements Specification. If the resulting animated specification will also pass the QACs required of operational software then fine, but it will in any case have served a valuable function.

The issue with regard to planning activity raised by Iteration is that planning must be iterative to be effective. Plans need continual re-adjustment in response to change of requirement, actual progress etc. Even if everything went entirely according to the original plan, there would still be a need to iterate planning since detailed planning of activities is not cost-effective until they are reasonably imminent. This means that plans roll, unfolding their detail as the project progresses. Both the Contractual and Actual Plans will exhibit this, but it is likely that the Actual Plan will hold more detail and its iterations will be more frequent

4.10 Re-usability

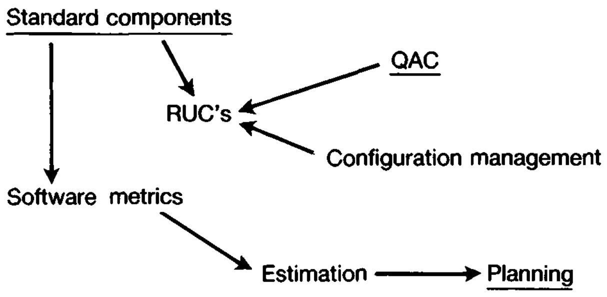


Fig. 16

One of the major expected benefits of IPSEs is extensive re-use of components with all the cost-saving that implies. It should be clear that if significant progress is to be made in this direction there must be standardisation of components, and ways of describing them and accessing them under configuration management. In addition, acceptable (no doubt very high) quality levels will be necessary, if a component is to be re-usable, so that rigorous QAC is also a prerequisite.

We have already covered the re-use of plans but there are other ways in which the experience of the past may be re-used. Once we have sufficient standardisation of components, *and not before*, we can start to collect meaningful software metrics. The fact that such a belief is more controversial than true may be largely because it is so controversial. Much of the data collected in the name of Software Metrics is of very doubtful value, except in those privileged development environments, which can already be highly controlled. Once standardisation has done its work, good metrics will not only have productivity uses but will also allow the development of valid estimation data, which will enable more accurate planning.

4.11 Automation

It was previously suggested that the introduction of COMMON SENSE rigour was similar to the cleaning up of manual stock records prior to computerising stock control. It is not an uncommon experience to find that such an improved manual system is superior, not only to the original system, but also to the proposed computer system. We are convinced that this is not the case with project environments. While the benefits of an IPSE should be great, the effort of maintaining one manually is prohibitive. Success with COMMON SENSE, as with structured methods, demands automation.

4.12 Decoupled issues

It was stated earlier that a number of important issues could be satisfactorily decoupled from COMMON SENSE. These have either not appeared at all, or only as a single unexpanded issue in the concept maps. To reduce the risk of scandalising the reader, some of those issues are mentioned here.

Estimation: The fact that standardisation should lead to more effective estimation was stated with no particular emphasis. It should, nevertheless, be recognised that many people have good cause to see improved estimates as the most important goal. It is significant that so much must be in place before it can reasonably be expected.

Enforcement of standards: This decouples as a management issue but with a small complication. Introduction of COMMON SENSE and IPSE is likely to demand closer adherence to even more standards, in environments where hardly anyone keeps to even the existing standards. Again automation is essential and must make adherence to standards the easy option.

Organisation: Theoretically, the choices of organisation for projects, and for the IS Shops which host them, are independent of COMMON SENSE.

Despite this it will soon be appreciated that some organisational structures will be more supportive of COMMON SENSE than others. We also take the view that a necessary part of developing a quality culture is that everyone should have some understanding of and concern for the whole process and for the maintenance and evolution of the environment. Practically no-one could be expected to be familiar with all the detail of SENSE, but everyone concerned in development should have COMMON SENSE.

5 The Software Factory

A guiding light for many concerned with IPSE has been the idea of the Software Factory. Software Engineering is seen as an attempt to achieve the same level of discipline for software development as is associated with the more mature Engineering disciplines employed in factories. We took scant regard of this idea in the development of COMMON SENSE and SENSE, feeling that software development is equivalent to engineering design rather than factory production and the former exhibits much the same lack of rigour that we were trying to avoid.

In retrospect, however, the factory concept can throw light on two issues. Firstly it is interesting that we are talking about factories They contain processes which yield factory output, and processes which maintain the factory. This is very different from the "do or die" project syndrome where piratical project managers are allowed to destroy the environment for any contemporary or future users Secondly it sheds some light on the way in which COMMON SENSE differs from most existing and conjectured IPSE products.

The idea of the Software Factory can be represented as an attempted correspondence between the elements of a factory and those of the software development support environment. For example:

Factory	↔	Support Environment
Work Order	↔	Contract
Stock Control	↔	Configuration Management
Machine Station	↔	Software Tool
Routing	↔	Standard Networks

Clearly the analogy cannot be taken too far but the role of Standard Networks in embodying methodology comes out. Most existing IPSE products attempt no more than "Stock Control" and "Machine Stations" by the provision of a Project Database under Configuration Management and the provision of interfacing aids to allow user-chosen software tools to be plugged in. None seem to have a formal way of embodying "Routing" such as we are suggesting can be done with standard networks.

6 Sense

As previously mentioned, SENSE is the environment which has been specified for GIS. It conforms to COMMON SENSE but details our choice of methodology and policy. For example, we have at the moment only one Standard Plan. This is aimed at large transaction processing database systems. It formalises our choice of "production" methodology which although embracing most of the techniques of LSDM, has strong leanings towards CACI. In planning we stipulate the use of Precedence Diagramming Method and, as already seen, our quality methodology includes Fagan Inspection.

Implementation and any necessary further detailing of the procedures is now being carried out by the implementation project. SENSE is very all-embracing so that this is no small task. Different managers from our applications development projects are taking on responsibility for implementing different pieces of methodology. This is controlled by contracts issued by the SENSE Installation Project Manager, who requires a good grasp of COMMON SENSE to see how everything will be welded together.

One of the sought-for benefits of COMMON SENSE is ease of future environment enhancement. As more and more of COMMON SENSE is put in place, it should become easier to develop standards and methodology. There is a danger that the SENSE installation project will become confused with the ongoing process of methodology/standards development. In order to avoid the "never-ending project" syndrome it is important that the installation project should only put in place enough to create an initial COMMON SENSE environment. For instance, in GIS a Standard Plan for Package Modification will soon be needed, followed shortly after by one for Expert Systems. Such developments should not be part of the initial installation since that aspect of COMMON SENSE is in place once we have one standard plan.

7 Conclusion

The reader who has reached this point may well have strong feelings about whether the name COMMON SENSE is justified. Whatever these feelings may be, we hope that this paper will stimulate serious efforts and cooperation towards achieving an acceptable common "view".

References

1 PHILIP VEASEY.: "The Project Process and its Support", An unrestricted internal ICl paper issued 27/11/85.
2 PHILIP VEASEY and STUART POLLARD.: "The Magic Word Contract", A paper currently being prepared for publication, available in draft form from the authors.
3 S.J. POLLARD and C.R. CRAWFORD.: An Approach to Information Technology planning. *ICL Technical Journal* pp.228–252 1986

Suggested further reading:

4 TONY DIGNAN.: "Strategy for Knowledge Based IPSE Development", Alvey Directorate, August 1984.
5 M.M. LEHMAN and N.V. STENNING.: "Concepts for an Integrated Project Support Environment" *Data Processing* Vol. 27, No. 3
6 ROBERT M. POSTON.: "Software Standards" *IEEE Software* August 1984.
7 BARRY W. BOEHM *et al.*: "A Software Development Environment for Improving Productivity, *Computer*, June 1984.
8 DAVID BROUGHTON.: "Building an IPSE From Proven Methodologies", *Data Processing*, Vol. 27, No. 3 August 1985.
9 "Draft Standard Taxonomy for Software Engineering Standards (P1002/D05)" *IEEE*, September 1985.

Global Language for Distributed Data Integration

P.M. Stocker
Computing Centre, University of East Anglia, Norwich, NR4 7TJ.

Abstract

It is now generally appreciated that it is desirable that the meta-data of the database should be accessible as data within a database model or schema. This paper discusses a system that allows this and shows how a given schema can be transformed into a particular modelling style, in this case NIAM. It is suggested that in a distributed database system the schema should be regarded as a transmissible standard data structure, rather than as a language for its definition.

1 Introduction

Database technology arose, more than twenty years ago, in response to the need to provide integrated sets of files and procedures. Today there is no single, standard database system; though the number of such systems has possibly reduced over the past five years. In view of the experience gained from programming languages, this variety seems likely to continue for years to come. The intention in the early days was to avoid separate data files being maintained by different sectors of an organisation. Now data or information is a more marketable commodity. For a variety of reasons the interchange of information within organisations has increased, and the flow between organisations will increase rapidly if technical facilities are improved.

Over the decade a great deal of attention has been given to the development of enhanced information systems, and to the improvement of methodologies for this purpose. This has led to proposals for many Conceptual Schema languages. Such languages, although often seen as part of the development of a system, are clearly equally useful as a description of its content when it has been established. There is a greater variety of conceptual schema languages than database systems, and it is often not clear how schema and implementation are to be related. Van Griethuysen[1] forms a useful reference on this field and on steps to achieve standards.

It seems likely that tools to achieve automatic or semi-automatic transformation of information from one system to another will have value for some time to come. This paper is based upon one such tool; the intention here is to show, by means of an example, the problems encountered and the degree of success achieved.

1.1 Mappings between representations

A great deal of the work carried out by the computer systems is the transformation of representations of information from one form to another. This transformation may be physical; one might at this point change the fount that is in use for this text. That would be a trivial but genuine change in the *physical* representation of this information. Alternatively, a sentence might be translated into French; which may be regarded as a *logical* transformation. In each of these cases one may assume that if the mapping is carried out sufficiently carefully no information is lost. There are circumstances where a mapping may desirably lose some information [Stocker[2]], and others where there is insufficient information in the source to meet fully the requirements of the target.

At the present time there is considerable emphasis on the concepts of knowledge, information and data. In the earliest days of digital computers results were often printed, via paper-tape or cards, as columns of numbers free from headers and any textual information. There will be agreement that these were data.

Subsequently, it became prudent to print headers to the columns and even to include the occasional paragraph of textual information. Most readers will feel that this output was still entirely data. The textual material was data which when interpreted by the reader became information. If two columns were printed, both marked *salary*, then it is possible that one might have found data within the computer system which revealed that these two headers were physically the same, but nothing to indicate that the items represented in the columns below were logically related in some way. As sophistication in data-processing has increased, the world of data dictionaries and conceptual schemas has been entered. In addition to the data, the system now holds meta-data, textual or structural, which is concerned with both the logical and physical representation of the data itself. In plainer English, it now contains data the object of which is to lead the observer into believing that the system understands what it is talking about. Or perhaps the reader may feel it does understand what it is talking about.

1.2 Roles of meta-data

The attitude taken in Stocker[2] is that it is useful to separate this meta-data into two classes. The first class is that data which can be used by the system to enforce the integrity of the information base. A satisfactory name for such data is integrity constraints, if one interprets this to include functional relationships between data-items, however complex. Such constraints are fundamentally logical, and most of these meta-data are concerned with logical concepts; such of the meta-data as are concerned with physical representation are invoked only at the lower levels of implementation or of data-transmission.

The second class of meta-data is that which serves to interpret output from the knowledge base to the human observer. It has no effect on the behaviour of the database system; it is added to the output from that system as colour is added to a painting book.

The author and his colleagues have been interested for some years in meta-data of the first of these classes. The collection of such data into an organised form for fairly simple cases was described briefly in Stocker[3], and more fully in Stocker[4]. The representation used is called the *abstracted conceptual schema*; 'abstracted' because some of the information in the conceptual schema, that of the second class, is not specifically captured in the mapping. This schema template, referred to as the ACS for brevity, is discussed in the following sections; it has been used to transform the model of one particular database system into that of another, but the emphasis in these cases [Akinyokun[5], Akinyokun[6]] has been structural, though Laender[7] is concerned with semantics

More recently the author became interested in the way which the content of a database might be presented on an open network in the circumstance where it was intended for use in conjunction with databases at other nodes, the *multi-database* of Litwin[8,9]. This interest is in the case where 'the user' of the databases is a computer system, not a person. In this situation there is a double requirement: first to present the semantics and content of the database to a human in order that the relevance and value to the application may be assessed; second, with human assistance, to present it to the nodal database system for logical integration, and for future use without further human intervention. There is a variety of forms that such an integration may take; the entirety of the relevant data might be moved, in which case the question of update frequency must be considered. At the other extreme no data may be moved, and the source site of the data interrogated remotely for each query. It is not intended to discuss this issue here, but it should be remembered as forming part of the environment in which the issues of this paper arise.

This paper presents the result of an investigation into how much semantic information is contained in, and can be extracted from, the integrity already present in the ACS. These constraints are primarily intended for use in maintaining data integrity; similarly it is assumed that the names that will be found in an ACS will be meaningful, but these will not have been chosen with the present purpose in mind. It is meant to be an appraisal of how the mapping would perform when applied to an existing database, with perhaps a small amount of extra effort. An ACS constructed specifically for network use would be considerably more informative. The investigation took the form of the development of a formal mapping from the ACS to a model of semantic-net type NIAM, [Nijssen[10], Verheijen[11]] and the computer implementation of this mapping. The details of the mapping are not presented here but the basis from which the semantic information is obtained is described.

2.0 The Abstracted Conceptual Schema

The original objective for which the ACS was designed was the specification of the functionality of and constraints upon the values of data-items contained within a database. This specification was sited at a level intermediate between a purely logical formulation and one in which the detailed physical layout is specified. In broad terms it specifies the data, but does not commit to, say, a relational or DBTG (i.e. CODASYL) implementation. The objective was to provide a standard form at a level intermediate between the Conceptual Schema and the Internal Schema, and to provide this in a way which is directly usable by a system implementor, and which is easily transportable between systems. It was used as the basis of a heterogeneous database system, Proteus, [Atkinson[12]], where the schema had to be a form which was easy to distribute to each of the nodes. In the literature, Conceptual Schemas have often been presented as languages, sometimes in conjunction with a transaction language, for example, DAPLEX, [Shipman[13]]. The ACS is not a language; it is a set of logical data-structures, which have simple associated semantics, which are to be used as a *target output* for Conceptual Schema languages, and as an *input* for database implementations. The structures are neutral in that they do not acknowledge, for example, entities or relationships, but are capable of representing these. The 'character' of a particular conceptual schema model will assert itself in the back-mapping from the ACS to the model. The structures are independent of any representation that is used for them. It is worth repeating that the structures store the meta-data, not the data. In a particular implementation it would be sensible to store the meta-data in the same structural representation as the data, probably in the same database system. The ACS structures are here represented in a relational form.

2.1 Components of the ACS

The ACS has three basic components; *data-item*, *element* and *value set*. Data-items have values, and are the only locations for the data. Elements are assemblages of data-items; they themselves have no names and are identified only by the values in the assemblage. Data-items would be used to contain the attribute values in an Entity-Attribute-Relationship schema; elements would be used for both entities and relationships. Data-items of like logical sort are given a type name called the data-item name; this typing is not related to traditional classifications such as text, integer, etc. An element is an assemblage of data-items; the association between a data-item and an element is specified. This association is called a *function*. An element may have more than one association with a data-item of the same data-item name, the distinction made by means of the function. Functions have names; different data-items may be associated with the same element by means of the same function name. The function/data-item mechanism is provided in the

ACS as a useful one for the database implementor working from a conceptual schema. The combination of function/data-item is referred to as a *property*.

2.1.1 Element sets All elements which have like associations (properties) can (and must) be combined into a set, called an element set, which has a name. It can be seen that where, as in this case, the element set is represented by a relation, then the properties form the column headers, while the data-item names classify the data-items into logical domains. We shall regard all the data as logically contained in such element set relations. Within an element set, the elements may be identified one from another by the value of one of the properties (or combination of values of a group of properties) which is called an *identifier*. An element set may have more than one identifier.

2.1.2 Value sets For the present purpose a value set can be defined as the set of values which comprise one column of one of the element set relations, with duplicates omitted, or as a generalisation it may be a set of pairs, triplets, etc. from an element set, a set of sub-tuples.

2.2 An example

Figure 1 illustrates pictorially the meta-data for a single element set, taken from the example which will be used throughout this paper. A university has a unique name and a unique code within the logical database. It has in addition a single address and a single telephone number.

The information displayed graphically here involves entries which occur in five relations in the ACS which contains the meta-data. The ACS is not

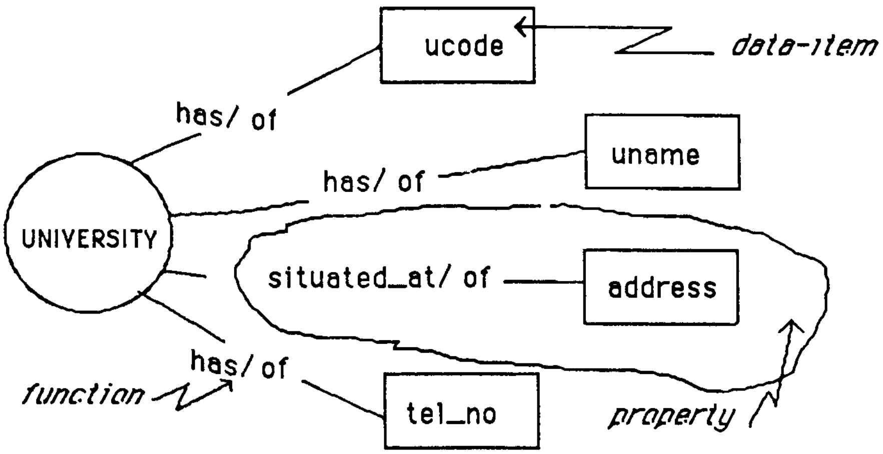


Fig. 1. The function is shown as a forward (entity → data-item) name and an inverse function name.

defined actually as shown here; names are converted from text strings to numeric identifiers at the earliest opportunity, but the strings are shown here for readability, and the equivalencing to numerical identifiers is omitted. With this modification the five relations of the ACS so far involved are:

ES_NAMES
FN_NAMES
DI_NAMES
ASSOCIATIONS
IDENTIFIERS

These may be seen in Appendix A. The ACS schema instance is that used in Nijssen[10]. The presentation adopted here is to show the whole of that schema, and to show in **bold** those parts which are specifically used to illustrate this text. Figure. 2 shows a corresponding illustration for another element set PERSON.

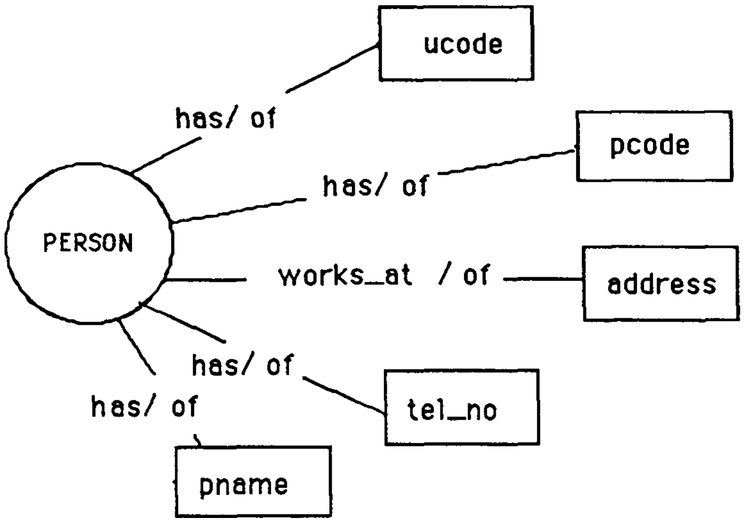


Fig. 2

This element set represents another entity; persons are associated with a university. This association is accommodated in the data by including the data-item *ucode* in the element set PERSON; *ucode* is an identifier in the element set UNIVERSITY and hence associates a person with a particular, single university. This way of representing associations is typical of relational databases and of any system where all structural associations are explicit. It can be noted from Appendix A that the identifier *pcode* is not regarded as unique overall, but only within a university. An identifier of person is, therefore, (*pcode*, *ucode*) and not *pcode* alone.

There are further elements in our specimen database. Universities offer degree courses; these are identified by the combination of (*ucode*, *dgcode*).

The degree courses are composed of lecture courses; a lecture course may be part of more than one degree course. Lecture courses are identified by the combination (*ucode*, *ccode*). These additional data-items cause further entries in the relations listed to above. The relationship between degree courses and lecture courses requires an additional element set, named CONTAINS.

2.2.1 The role of Value sets The use of value sets to express constraints can be seen in Appendix A as follows. All the element sets PERSON, DEGREE_COURSE and LECTURE_COURSE involve the data-item *ucode*, the elements have meaning only within the context of a particular university. It follows that the *ucode* associated with each element in these sets must be one of those belonging to the universities in the element set UNIVERSITY. In other words, the set of values for *ucode* for each of these element sets must be contained within the set of values for UNIVERSITY. Similarly, the set of values for the combination *ucode*, *ccode* in CONTAINS must be included within the set derived from LECTURE_COURSE, because a degree cannot contain a lecture course which does not exist. There is a corresponding constraint on *ucode*, *dgcode*. These constraints are shown in Appendix A in the relation named CONSTRAINTS; they are the traditional data validation conditions.

3.0 The NIAM schema

NIAM is a schema language related to semantic nets; it provides an implicit description of the data relationships rather than an explicit one. There are two basic types of object in the meta-data, lexical objects and non-lexical ones. Non-lexical objects, as for ACS elements, have no name and cannot be written down; they are identified by the attributes which are associated with them. The attributes are called lexical objects because they are names, and can be written. Non-lexical objects of common form are grouped as a type, a non-lexical object type of NOLOT. The lexical objects are grouped as LOTS. The relationship between a NOLOT and a LOT can be shown diagramatically as in Fig. B1. PERSON is a NOLOT and *pname* is a LOT. The symbol

```
<------->
|_with___|__of__|
```

in conjunction with, for example, PERSON and *pcode*, means that there is only one *pcode* for each person, but that there may be more than one person with the same *pcode*.

```
<------------->
|______|______|
```

is the symbol which signifies an M to N relationship between two objects, while

```
<------------><------------>
|____________|_____________|
```

denotes a one to one relationship.

Most of the LOTS have been omitted in the Fig. B1 in order to maintain clarity. They are identical to the DI-names (since in each case they are data-holding elements) and are relatively unimportant here because the relationship LOT and NOLOT does not differ from that between data-item type and entity set. It is the inter-relationships between NOLOTS which are of interest.

4.0 The mapping

It is clear by inspection of the relation ES-NAMES of Appendix A and the table of NOLOTS in Appendix B, that the majority of element sets map direct into NOLOTS. In the ACS there is no specified connection between element sets; connections are implicit through common data-items. If the element sets of the ACS are treated simply as relations, then one is free to join them on any compatible data-items. Not so in NIAM where queries must follow the specified logical access paths. The join of Lecture-course to Person on *ucode* is meaningless, but proper in the relational system. It is plain on inspection of Appendix B that a LOT may occur once and once only in NIAM, whereas a data-item type may occur many times in the ACS. The resolution of this difference accounts for most of the structural differences between the ACS and NIAM.

4.1 Element sets which map into NOLOTS

It is part of the ACS semantics that for every data-item which occurs as an identifier there must be one and only one value set which contains all its values, though the same values may occur in combination elsewhere. It is not required, however, that this set shall be explicitly stated for each data-item. These sets may be deduced from the value set constraints if the ACS is correctly posed. Where a data-item occurs other than in its defining set it will take part in a NIAM link and will not appear as a LOT. The data-item appears as a LOT in a NOLOT corresponding to the element set which defines it. *Ucode* occurs in both Fig. 1 and Fig. 2. *Ucode* occurs in the NOLOT University_Poly, but not in the NOLOT Person; both take part in the link from University-Poly to Person.

4.2 Element sets which do not map into NOLOTS

These sets may be illustrated by the element set Contains. This set has three properties with data-items *ucode*, *dgcode* and *ccode*; *ucode* and *dgcode* are contained within the values for the element set Degree-Course. They would form part of a linkage from a NOLOT Contains to the NOLOT Degree-Course. A similar statement holds for Contains, *ucode*, *ccode* and the NOLOT Lecture-Course. It can be seen that as a result of constructing these two linkages the potential NOLOT Contains now has no LOTS. It serves no purpose and only the link is required. In an obvious language the element set becomes a relationship and not an entity. It is worth noting that if the element set Contains were given some additional trivial property, for example a flag indicating whether the degree-course contains the lecture course every year or only some years, then the element set would lead to an entity like NOLOT and not a relationship. The author suggests that this may indicate that there is no deep philosophical distinction between entities and relationships within a relational data environment, because data specification and relationship specification are mixed together in the same relation. There is a more clear distinction in a logical modelling environment with implicit realisation of the linkages (e.g. NIAM).

In this particular case it may be worth noting that the ACS condition says more than the NIAM drawing; though the NIAM definition language is more powerful. The ACS element set requires that a lecture course may form part of the degree course in the *same* university. The NIAM drawing requires only that degree courses should be composed of lecture courses. That is, the Paris degree course could include lectures from Lyons.

4.3 NOLOTS which have no corresponding element set

It was stated earlier that a LOT may occur only once in NIAM. There are several data-items which occur more than once in the ACS; one of these is *address*. In order to meet the NIAM requirement it is necessary to generate a NOLOT, Addresses, which has a single associated LOT, address.

In the context of relational operations upon the data (as distinct from the metadata) it is not clear what purpose is served by this element set. It would be valuable to ask 'Who is the owner of this address', but that can only be compiled if we know the names of all the NOLOTS which are connected to the NOLOT Address. Thus this particular type of NOLOT appears to have value only where data and metadata are simultaneously available. This may be the place for the author to declare that he believes that the distinction between data and metadata should not be made, and that any necessary formalisations required to make relational languages deal with dynamic relation names from within the transaction should be carried out, together with the necessary modification to the database system. If that were done then this type of NOLOT would be of much more value.

4.3 Other NOLOTS with a single associated LOT

A reader with a sound knowledge of NIAM will already be objecting that the concepts of LOT and NOLOT have been inadequately presented here. A LOT is meant to be the name of something. To elaborate on this Degree-Type will be discussed. In the original version of this schema instance this appeared as a simple property. In the automated mapping as it stands at present that would produce a LOT in the NIAM output. In fact, human interpretation from an experienced NIAM user requires a NOLOT with a single LOT (NSL). This differs from the previous case in that it has only one linkage also. This, if physically implemented, leads to a file in which one presents the key in order to obtain the key.

The example shown has the same superficial structure for a different, but perhaps not unrelated reason. There is often an integrity constraint on the allowed values for such a data-item. In this instance an element set Degree-types holds the allowed values B.Sc, B.A., M.Sc and so on. Two other elements, Statuses and Sexes reflect similar situations. Another possibility is to use a NSL for any data-item which does not occur in any identifier.

5 Names

If the mapping of an ACS-like description into a local conceptual model, possibly with associated graphics, is to be the second level for network database assessment (one assumes that the first level would be totally content rather than structure oriented), then function and data-item names must carry the semantic load. At the next level one is likely to encounter more detailed textual accounts of functions and data-items.

The names in this schema instance were deliberately chosen to be a rather mixed bunch, some informative, many not. The content of the NIAM drawing was produced automatically, but not the drawing itself. The output from the mapping program is close to that shown in Appendix B. It is thought that the output is readable; structurally the output matches that produced by a human for the same problem. In this and other mappings included in the references there are situations where the source has more names than are required in the target. The ACS has redundant naming for properties except in the instances where a data-item occurs twice in the same element. Equally there may be a situation where the target requires more names than are present in the ACS. The latter situation occurs twice in this mapping. First it occurs in the case where element sets reduce to relationships; the only useful name available is 'Contains'; this is the wrong part of speech and must serve both forwards and backwards. Those interested in language may say that suitable forms can be constructed, but it must be remembered that we are dealing with a free situation, and the database constructor may not have used a third-person present for the element set name. The aim is an ad hoc mechanism to produce a moderately intelligible first result. The second form of constructed names are those corresponding to NSLs, though this problem disappears if identical LOT and NOLOT names are permitted. In reality the only test of how much semantics can be extracted from the names will be actual usage of real examples.

Over the next decade, with increasing use of logic languages, networks and commercial information services, it seems likely that the role of simple phrases in information systems will become more controlled and formalised.

Acknowledgements

The author wishes to acknowledge the co-operation of F. Van Assche in constructing the mapping and for the provision of NIAM expertise and skills. He is not responsible for any of the deficiencies in the paper. The author wishes also to acknowledge that the paper arises from a part of the work of a project supported by the EEC under the MAP programme.

References

1 VAN GRIETHUYSEN: Concepts and Terminology for the Conceptual Schema and the Information Base. ISO TC97/SC5/WG3.

2 STOCKER, P.M.: Canonical Schemata, Canonical Queries and Standardisation, in DATABASE, *Infotech State-of-the-Art Report*, Pergamon Infotech, (London 1981).

3 STOCKER, P.M. and CANTIE, R.: A Target Logical Schema: The ACS, in Proceedings of the 9th International Conference on Very Large Data Bases, Florence, 1983. (North-Holland, 1983).

4 STOCKER, P.M. and CANTIE, R.: A Target Logical Schema: The ACS. University of East Anglia, School of Computing Studies and Accountancy – Database Research Group Internal Report No. 1, 1983.

5 AKINYOKUN, O.C.: A Methodology for the Automatic Design of Database Systems. Ph.D Thesis, University of East Anglia, School of Computing Studies and Accountancy 1984.

6 AKINYOKUN, O.C. STOCKER, P.M. and BORGES, M.R.S.: Bi-directional Mapping between a User-oriented Conceptual Schema and a Target Logical Schema: The ACS, in Proceedings of the 4th British National Conference on Databases (BNCOD4), July 1985. (Cambridge University Press, 1985).

7 LAENDER, A.H.F. and STOCKER, P.M.: An Interactive Database End User Facility for the Definition and Manipulation of forms, in Research and Development in Information Retrieval, Proceedings of the Third Joint BCS and ACM Symposium, (Cambridge University Press, 1984).

8 LITWIN, W. and KABBAJ, K.: Multidatabase Management Systems. International Computing Symposium 1983, Nurnberg.

9 LITWIN, W.: MALPHA: A Multidatabase Manipulation Language. EUTECO, Varese, 1983. (North-Holland 1983).

10 NIJSSEN, G.M.: On the Gross Architecture for the Next Generation Data Base Management Systems, in Information Processing, Proceedings of the 1977 IFIP Congress, Toronto. (North-Holland 1977).

11 VERHEIJEN, G.M.A. and VAN BEKKUM: NIAM: An Information Analysis Method, in Information Systems Design Methodologies. Proceedings of the IFIP TC8 Working Conference, Noordwijkerhout, 1982. (North-Holland 1982).

12 ATKINSON, *et al.*: 'PROTEUS': A Heterogeneous Distributed Database Project, in DATABASE – Role and Structure (Editors Stocker, Gray and Atkinson) (Cambridge University Press, 1984).

13 SHIPMAN, D.: The Functional Data Model and the Data Language DAPLEX. SIGMOD 79, Boston, Massachusetts 1979.

APPENDIX A

ES-NAMES

UNIVERSITY_POLY
DEPARTMENT
PERSON
STAFF
DEGREE_COURSE
LECTURE_COURSE
SEXES
STATUSES
DEGREE-TYPES
OFFERS
RUNS
COURSE-PREREQUISITE
CONTAINS

DI-NAMES

UCODE
DG-CODE
UNAME
DTITLE
ADDRESS
TEL-NO
PNAME
DNAME
FAC-SCH-NAME
SEX
PRENOM
TEL-EXT
STATUS
CCODE
YCODE
CTITLE
DEGREE-TYPE
DEPT-CODE
PCODE
UCCA-CODE

FUNCTION NAMES

HAS	**OF**
SITUATED_AT	**OF**
WORKS_AT	**IS_WORKPLACE_OF**
IS_WITHIN	INCLUDES
HOLDS	IS_HELD BY
MAY_HAVE_VALUE	IS_ALLOWABLE_VALUE
HEADED_BY	IS_HEAD_OF
NEEDS	IS_NEEDED
CO_ORDINATED_BY	CO_ORDINATES

ASSOCIATIONS

UNIVERSITY-POLY	**HAS**	**UCODE**
	HAS	**UNAME**
	SITUATED-AT	**ADDRESS**
DEPARTMENT	HAS	UCODE
	HAS	DEPT-CODE
	HAS	DNAME
	HEADED-BY	PCODE
	IS-WITHIN	FAC-SCH-NAME
PERSON	**HAS**	**UCODE**
	HAS	**PCODE**
	HAS	**PNAME**
	HAS	PRENOM
	SITUATED-AT	**ADDRESS**
	HAS	TEL-NO
	HAS	SEX

STAFF	HAS	UCODE
	HAS	PCODE
	WORKS-AT	ADDRESS
	HAS	TEL-EXT
	HAS	DEPT-CODE
DEGREE-COURSE	HAS	UCODE
	HAS	DG_CODE
	HAS	DTITLE
	HAS	DEGREE-TYPE
	HAS	UCCA-CODE
LECTURE-COURSE	HAS	UCODE
	HAS	CCODE
	HAS	CTITLE
	HAS	YCODE
	COORDINATED-BY	PCODE
SEXES	MAY-HAVE-VALUE	SEX
STATUSES MAY-HAVE-VALUE	STATUS	
DEGREE-TYPES	MAY-HAVE-VALUE	DEGREE-TYPE
OFFERS	HAS	UCODE
	HAS	DEPT-CODE
	HAS	DG CODE
RUNS	HAS	UCODE
	HAS	CCODE
	HAS	DEPT-CODE
COURSE-PREREQUISITE	HAS	UCODE
	HAS	CCODE
	NEEDS	CCODE
CONTAINS	HAS	UCODE
	HAS	DG-CODE
	HAS	CCODE

IDENTIFIERS

UNIVERSITY-POLY	HAS	UCODE
	HAS	UNAME
DEPARTMENT	(HAS	DEPT-CODE
	(HAS	UCODE
PERSON	(HAS	PCODE
	(HAS	UCODE
STAFF	(HAS	PCODE
	(HAS	UCODE
DEGREE-COURSE	(HAS	DG-CODE
	(HAS	UCODE
LECTURE-COURSE	(HAS	CCODE
	(HAS	UCODE
OFFERS	(HAS	UCODE
	(HAS	DEPT-CODE
	(HAS	DG_CODE
RUNS	(HAS	UCODE
	(HAS	DEPT-CODE
	(HAS	CCODE

COURSE-PREREQUISITE	(HAS (HAS (NEEDS	UCODE CCODE CCODE
CONTAINS	**(HAS (HAS (HAS**	**UCODE DG-CODE CCODE**

The (indicates that the properties are to be taken as a group to form the identifier.

VALUE SETS

CONTAINED SET	CONTAINING SET	SET VALUE
PERSON	**UNIVERSITY-POLY**	**UCODE**
DEPARTMENT	UNIVERSITY-POLY	UCODE
DEGREE-COURSE	**UNIVERSITY-POLY**	**UCODE**
LECTURE-COURSE	**UNIVERSITY-POLY**	**UCODE**
STAFF	**PERSON**	**UCODE PCODE**
STAFF	DEPARTMENT	UCODE DEPT-CODE
DEPARTMENT	STAFF	UCODE PCODE DEPT-CODE
OFFERS	DEPARTMENT	UCODE DEPT-CODE
OFFERS	DEGREE-COURSE	UCODE DG-CODE
RUNS	DEPARTMENT	UCODE DEPT-CODE
RUNS	LECTURE-COURSE	UCODE CCODE
CONTAINS	**DEGREE-COURSE**	**UCODE DG-CODE**
CONTAINS	**LECTURE-COURSE**	**UCODE CCODE**
COURSE-PREREQUISITE	LECTURE-COURSE	UCODE HAS/CCODE
COURSE-PREREQUISITE	LECTURE-COURSE	UCODE NEEDS/CCODE

APPENDIX B

NOLOT NAMES

UNIVERSITY-POLY
DEPARTMENT
PERSON
STAFF
DEGREE-COURSE
LECTURE-COURSE
SEXES
STATUSES
DEGREE-TYPES
ADDRESSES
TELEPHONE

LOT ASSIGNMENTS

UNIVERSITY-POLY	HAS	UCODE
UNIVERSITY-POLY	HAS	UNAME
PERSON	HAS	PCODE
PERSON	HAS	PNAME
PERSON	HAS	PRENOM
STAFF	HAS	TEL-EXT
DEGREE-COURSE	HAS	DG-CODE
DEGREE-COURSE	HAS	DTITLE
DEGREE-COURSE	HAS	UCCA-CODE
LECTURE-COURSE	HAS	CCODE
LECTURE-COURSE	HAS	CTITLE
LECTURE-COURSE	HAS	YCODE
SEXES	MAY-HAVE-VALUE	SEX
STATUSES	MAY-HAVE-VALUE	STATUS
DEGREE-TYPES	MAY-HAVE-VALUE	DEGREE-TYPE
DEPARTMENT	HAS	DEPT-CODE
DEPARTMENT	HAS	DNAME
DEPARTMENT	HAS	FAC-SCH-NAME

LOT LINKAGES N --> 1

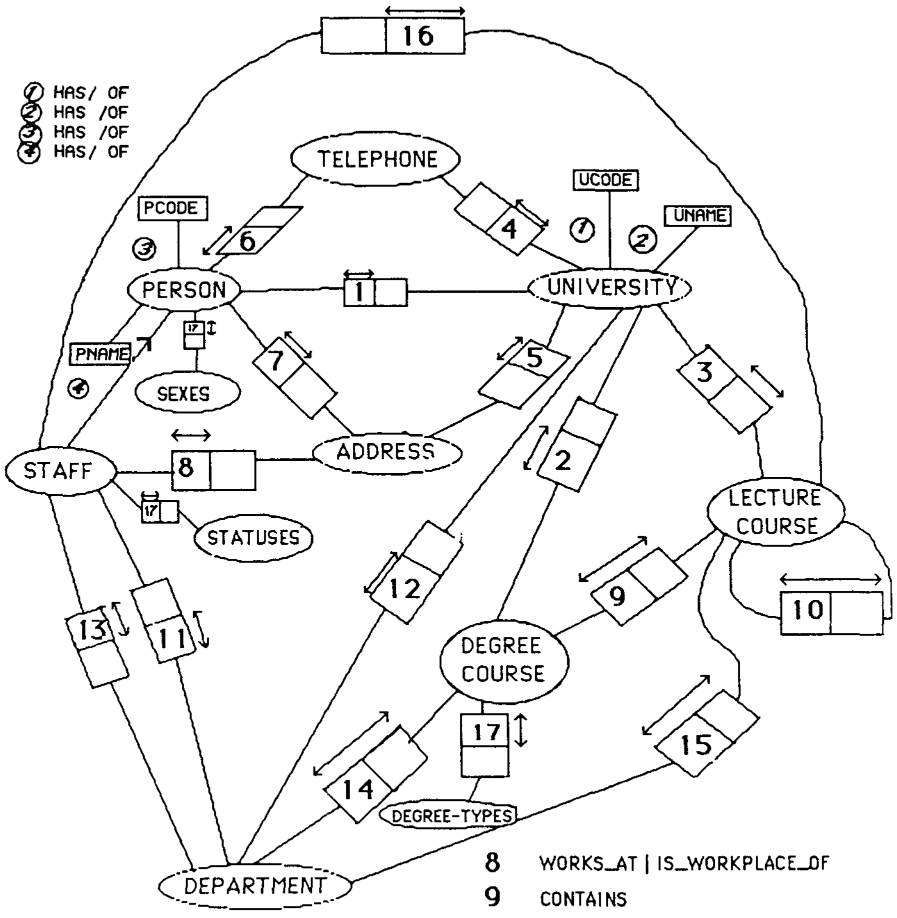
DEPARTMENT	OF	UNIVERSITY-POLY
PERSON	OF	UNIVERSITY-POLY
DEGREE-COURSE	OF	UNIVERSITY-POLY
LECTURE-COURSE	OF	UNIVERSITY-POLY
STAFF	OF	DEPARTMENT
LECTURE-COURSE	IS-COORDINATED-BY	STAFF
DEPARTMENT	HEADED-BY	STAFF
PERSON	HAS	SEXES
STAFF	HAS	STATUSES
DEGREE-COURSE	HAS	DEGREE-TYPES
UNIVERSITY-POLY	SITUATED-AT	ADDRESS
PERSON	HAS	ADDRESS
STAFF	WORKS-AT	ADDRESS
UNIVERSITY-POLY	HAS	TELEPHONE
PERSON	HAS	TELEPHONE

NOLOT LINKAGES M --> N

DEGREE-COURSE	**CONTAINS**	**LECTURE-COURSE**
DEPARTMENT	RUNS	LECTURE-COURSE
DEPARTMENT	OFFERS	DEGREE-COURSE
LECTURE-COURSE	COURSE-PREREQUISITE	LECTURE-COURSE

ROLES

STAFF	**IS-A**	**PERSON**



1	HAS / OF
2	HAS / OF
3	HAS / OF
4	HAS / OF
5	SITUATED_AT | OF
6	HAS / OF
7	HAS / OF
8	WORKS_AT | IS_WORKPLACE_OF
9	CONTAINS
10	COURSE_PREREQUISITE
11	HEADED-BY / IS-HEAD-OF
12	HAS / OF
13	HAS / OF
14	OFFERS
15	RUNS
16	COORDINATED-BY / COORDINATES
17	MAY-HAVE-VALUE / IS-ALLOWABLE VALUE

Fig. B1

The design of distributed secure logical machines

R.W. Jones
ICL Defence Technology Centre, Winnersh, Reading, Berkshire

Abstract

The paper describes a model for secure distributed systems. It is based on the idea that such a system is to be regarded by its user as a logical machine and that the distribution of its parts is not necessarily known to him. The parts (called here *resources*) have *rights* to enable them to affect each other. Three basic *rights* are described: to *control*, to *authorise*, and to *send*. The *authorise right* enables its possessor to supply rights to the resource specified by the *right* but does not allow other control operations. The use of these rights to build secure systems from their basic elements is described.

1 Introduction

This paper describes work done by the author at International Computers Limited as a contribution to an architecture for data processing systems which use *resources* which may be remote from each other, accessed by means of telecommunication lines. The underlying idea is that such a distributed system can be regarded as a logical machine. This is an extension of the idea of a virtual machine within a single computer installation. The resources which form a logical machine have their relationships defined as explicit *rights*. There are defined operations which create and supply rights. The possession of and ability to operate on particular rights make the responsibility for the security of a logical machine explicit, ensuring that it can be constructed, modified and used only by those authorised to do so.

The objects and operations described form a model which is pragmatic in that it derives from what are believed to be the needs of practical systems. The intention is to refine and add to the model by continued comparison with existing systems and by comparison with other models.

The purpose of the work is twofold: on the one hand to provide a model against which real systems may be checked; on the other hand to describe objects and operations which may be implemented as the basis of secure distributed systems.

1.1 Definitions of terms

In the following, special or restricted meanings are given to a number of words that are in common use. These terms are defined when they first appear in the text, and are given there in italics; afterwards they are given in roman. The paragraphs in which they first appear, and are defined, are as follows.

activate	3.2.3
authorise	3.2.3
change	3.2.3
code	3.2.1
community management	3.5
control	3.2.3
control right	3.2.3
controller	3.3
basic controller	3.3
delete	3.2.3
end user	2
management entity	3.3
resource	1, 3.2.1
basic resource	3.3
creation resource	3.3
right	1, 3.2.3
actual right	3.2.3
formal right	3.2.3
send	3.2.3
send message	3.2.3
state	3.2.1
supply	3.2.3
suspend	3.2.3
type	3.4
withdraw	3.2.3

The sections of the paper discuss the subject as follows.

Section 2 describes the context, the provision of secure data processing systems based on *resources* accessible via telecommunication lines.

Section 3 describes the model in terms of its objects and the operations upon them. It also relates the objects and operations to those needed to establish a secure community of computer installations.

Section 4 gives examples of use of the model.

Section 5 makes some analogies with human based systems to justify the claim that the concepts are intuitively reasonable and useful.

Later papers will consider the following:

- Further examples of use of the concepts;
- The representation of rights in real systems;
- The relationship of the model to systems and programming languages which allow confined environments to be created and access to them to be regulated;
- The relationship of the model to other secure system models and to policies for information flow.

2 The context

The context is a community of computer installations (end systems in the terms of Ref. 1) which communicate using telecommunications and agreed standard protocols, in particular the Open Systems Interconnection standards associated with Ref. 1. Those standards allow a single end system to support a number of software entities and allow an individual entity in one end system to exchange messages with an entity in a remote end system. The communication path between the two may be a local area network, a wide area network or more than one network of either kind with appropriate gateways. This is illustrated in Fig. 1. The ideas concerning security described in the paper do not depend upon a particular set of communication standards. Rather they assume that some suitable communication medium is available. A network using OSI standards is a suitable example.

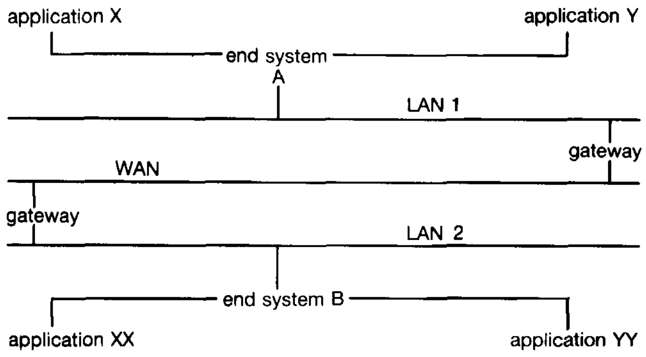


Fig. 1

A secure distributed system of the kind considered is then a network of communicating entities in one or more end systems. Such a distributed system must be available only to its authorised users, who must themselves be capable of only authorised actions and must be unable to interfere with, use or discover information about other systems or their users which share the same end systems and from which they are excluded.

Figure 2 illustrates a simplified system based on three end systems, labelled A, B and C. Each end system, whose boundary is shown in the figure by a

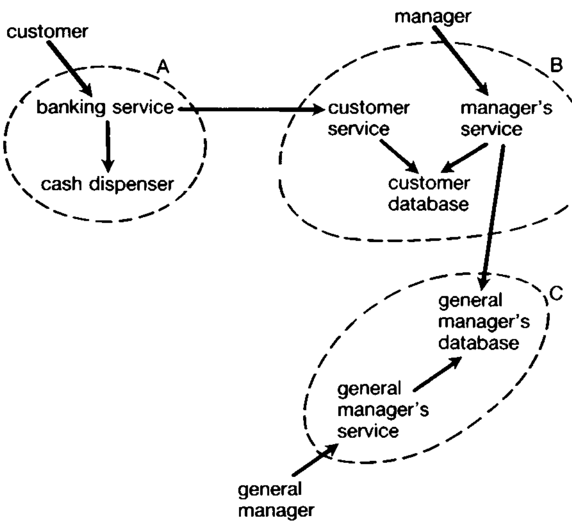


Fig. 2

dotted line, contains several logical entities – for example 'customer service' in B. The arrows show permitted communication lines between the entities. The items not within dotted line areas represent *end users* (people) who communicate with the computer system using some terminal device. The direction of an arrow shows that the entity pointed at is used by the one at the tail if one relates the diagram to the normal human perception of the work to be done. Thus the customer uses the banking service. The banking service uses the cash dispenser to provide the customer with money and uses customer service (via a telecommunication line) to check the customer's authenticity and to relate his request for cash to the state of his account.

Figure 2 illustrates also that in the context we are considering there may be overlapping rights to use resources. Thus 'Customer's Service' and 'Manager's Service' are both authorised to use 'Customer's Database'; 'Manager's Service' and 'General Manager's Service' are both authorised to use 'General Manager's Database', the former, presumably with more restrictions than the latter.

Figure 3 illustrates the fact that the end users to be considered are not only those who use the applications services provided by software running on the basic equipment. They include all those who need to interact with the equipment and its software to provide and support the application services.

Figure 3, thus, shows, in addition to the application user at installation A, an installation manager, responsible for providing the services at A and regulating their use, an engineer, whose access should be restricted to functions to enable him to check and maintain the correct operation of the installation, and a security manager using a resource to which security

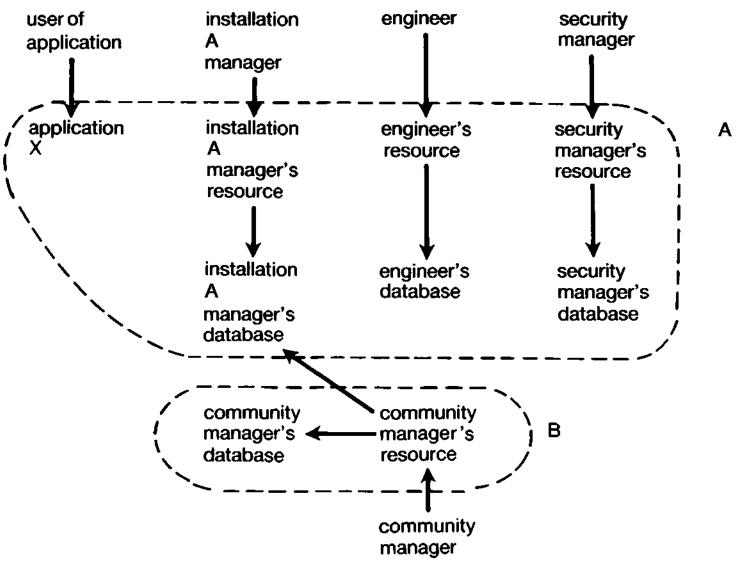

user of application
installation A manager
engineer
security manager
application X
installation A manager's resource
engineer's resource
security manager's resource
A
installation A manager's database
engineer's database
security manager's database
community manager's database
community manager's resource
B
community manager

Fig. 3

violations and other incidents of interest to security are logged. Installation B contains a resource which controls the allocation of installations to a community and the allocation of the resources the installations provide to each other. The person called the community manager has the right to this resource. One would expect also other resources and their users at installation B, not shown in the figure. In practice some of the people may have more than one role. Thus the community manager may also be the installation manager at B. The figure illustrates divisions of responsibility which the network may be required to keep separate and which the model should be able to describe.

3 The model

3.1 General

The model is described in term of its objects and operations. Words used in a special sense described in this section are in italics when they first appear.

3.2 Objects and operations

The model is a network of resources and end users. Each resource communicates with none, one or more other resources and end users. Each end user communicates with just one resource. This is shown in Fig. 4.

3.2.1 A resource A *resource* is a logical part of a computer system. It has *code*, a set of *rights*, and a *state* (see Fig. 5). Its code specifies what changes of state it will undergo and what messages it will send in all circumstances. Its

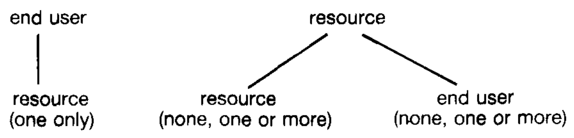


Fig. 4

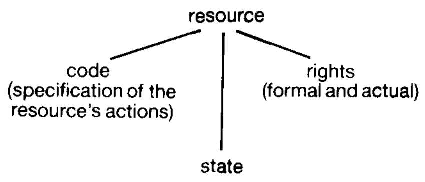


Fig. 5 A resource

rights specify which other resources and which end users it is allowed to affect. Its *state* represents its memory, including which parts of its code it must next obey and its data variables. Code, rights and state are described in more detail in 3.2.2, 3.2.3, and 3.2.4 respectively.

3.2.2 Code *Code* is a set of instructions obeyed by an interpreter (the interpreter being implicit in the definition of a resource). An example of an interpreter is a hardware CPU provided as basic installed equipment. Another is a software implementation which itself uses a CPU. An instruction may change the state of its own resource directly. If it is to send a message to or change the state of another resource or an end user it refers to one of the rights held by its own resource.

3.2.3 Rights A *right* is analogous to an external reference in a programming system which enables separately compiled modules of code to be used together. In this model, however, rights are always held separately from code. One effect of this is that the power of a resource to affect or communicate with other entities may be seen without examining its code.

A resource has *formal rights* and *actual rights.* The formal rights are a set of names, each representing a single right. They are used by the code of the resource to refer to its rights. Each actual right represents the ability of the resource to operate upon another specified resource or end user in a way defined by the type of the actual right. An actual right is supplied to the resource by an operation which defines which formal right it is bound to.

There are three types of right: *control*, *authorise* and *send.*

Control confers the ability to put the resource into service, withdraw it from service, and modify its function. There is only one holder of the *control right* of any one resource. This type of right is marked 'c' on diagrams.

Authorise confers the ability to provide the resource with rights already held by the provider (except control rights). This type of right is marked 'a' on diagrams.

Send confers the ability to send a message to the resource. This type of right is marked 's' on diagrams.

The rights are hierarchical in that the right to control a given resource includes all the abilities conferred by the right to authorise it and the right to authorise it includes all the abilities conferred by the right to send to it.

The operations for which a right is needed and their relationship to types of right are shown below.

One of the operands of each operation is a message which is delivered to the target resource. A code procedure specified for that operation at that resource is performed.

i *Operations permitted with control right*

Activate

This makes the resource available to holders of rights other than control. The code invoked initialises variables as necessary to present the resource in its initial state.

Suspend

This makes the resource unavailable except to the holder of its control right. The code invoked interacts with variables representing use of the resource to provide graceful suspension.

Change

This changes the resource's code. The message passed to the resource contains the change specification. The code invoked typically makes information available to other holders of rights to the resource concerning the change as well as making the change itself.

Delete

This withdraws the resource from service. The code invoked performs any necessary clearing up operations.

ii *Operations permitted with control or authorise right*

Supply

This supplies to the target resource a right, held by the supplier, of type 'authorise' or 'send'. When the operation has been successfully performed both the supplier and the target resource have the right. The message passed to the resource associates the actual right supplied with one of the resource's formal rights. The code invoked typically interacts with variables representing use of the resource to provide an orderly change to the resource as seen by holders of rights to access it. A particular authorise right may restrict the set of formal rights available to a subset of the total associated with the resource.

Withdraw

This withdraws from the target resource a right previously supplied by the resource which is withdrawing it. The message passed to the resource identifies the right. The procedure invoked typically interacts with variables representing use of the resource to provide an orderly change to the resource as seen by holders of rights to access it.

iii *Operations permitted with any type right*

Send message

This sends the message to the resource to be dealt with by the procedure representing the resource's application.

3.2.4 State Whereas the code and rights of a resource describe its potential, its state describes its current condition. It records which instruction of its code it must next obey. It is likely to record a stack of instruction pointers because procedures which are part of the code have been invoked in nested fashion.

A resource, as so far described, is what is often called a process with the distinguishing feature that rights are held separately from code.

It is likely that instructions of the resource give rise to independent asynchronous processes as part of the same resource and therefore subject to the same rights. The existence and state of these processes is part of the total state of the resource.

The data variables of the resource are a further part of its state.

3.3 Special resources

The following are distinguished.

i *Management entity*

This models all those human beings who install and control the basic computer hardware and software and authorise people to access it. Within the *management entity* there are relationships similar to those described in this paper. They are not, however, dealt with here.

ii *End user*

This models a human being who uses the computer system or in some way interacts with it, but is not part of the management entity. In the terms described here it is characterised by the fact that no resource has control right to operate upon it and by the fact that it is not created within the system modelled. In most of the discussion in this paper the term 'resource' does not include an end user.

iii *Basic controller*

This models the basic control software of a computer installation. In the terms described here it is characterised by the fact that its *controller* (i.e., the entity which holds the right to control it), is the management entity and that it is given the right to create resources (see v., below).

iv *Basic resource*

This models a facility installed as part of a computer installation, for example the set of operations provided by an operating system to enable users to create and use data files. In the terms used here a *basic resource* is characterised by the fact that the management entity is its controller and by the fact that it is not created by '*create resource*' (see 3.4). It is installed securely as hardware and software by means not described here.

v *The creation resource*

This is a special basic resource. A resource which has 'authorise' right to it may create another resource. One which has not cannot.

Figure 6 illustrates a computer installation, its resources and end users in the terms described here.

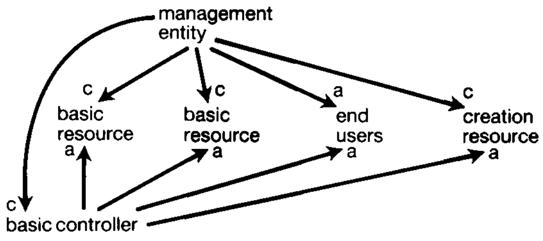


Fig. 6 Rights in a computer installation

3.4 Creation of resources and rights

The logical minimum is a management entity with authorise right to a creation resource. The resources created in practice by non-automated means are described in 3.3 and illustrated in Fig. 6.

A resource which has authorise right to the creation resource may use the operation 'create resource' to create a new resource. The form and meaning of the operation are as follows.

Create resource (code, rights)

The parameter 'code' is the code of the resource created. It includes an initial procedure, which is obeyed when the procedure is created, and the procedures corresponding to the operations described in 3.2.4.

The parameter 'rights' is the set of formal rights, referred to by the code, to which actual rights will be assigned.

The operation 'create resource' returns a control right to the created resource. The caller may assign this to one of its own formal rights.

In order to provide lesser rights (authorise and send) there must be some way of creating them. There is therefore an operation 'create right' whose form and meaning are as follows.

Create right (right, type)

The parameter 'right' is a right, held by the caller, whose type is 'control'.

The parameter 'type' is 'authorise' or 'send'.

The result is a right to operate upon the resource indicated by the 'right' operand with the type of the 'type' operand. It may be assigned to one of the caller's formal rights (and thence supplied to other resources as necessary).

The operations 'create resource' and 'create right' together with the special resources described in 3.3 enable a control hierarchy of resources to be created and a network of authorise and send rights to be distributed within it.

Section 3.5 describes how this network may be established in a community of end systems.

3.5 Creation of a community of end systems

We need to be able to control the transfer of rights among an arbitrarily large number of end systems. One way of achieving this is to support one of the end systems as the *community management system* (*CMS*) and to install the basic equipment so that basic controller of each of the other systems has the right to authorise the CMS and vice versa as shown in Fig. 7.

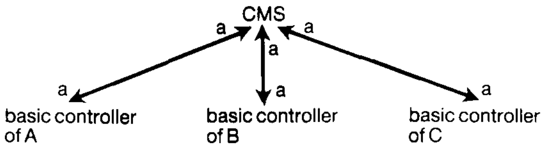


Fig. 7

Strictly speaking a right in only one direction is needed on installation since this enables the end which possesses it to send the reciprocal right.

In a very large community a hierarchy of CMS's may be created to provide local control of rights.

Since it is the management entity which controls all the basic controllers which represent the end systems of the community, including the CMS, the operations to create them and to provide the initial rights between the CMS and each of the basic controllers cannot take the form of calls upon software procedures which correspond exactly to 'create resource' and 'supply'. They are instead operations by installation staff which have the same effect.

The actions of the staff are affected by the way rights are represented, a topic not so far dealt with in this paper. It must be such that the possessor of a right does not abuse it (for by creating from it a more powerful right to the same resource) and gives it up when the 'withdraw' operation is performed. It is intended to deal with the topic in more detail in a later paper. Within an end system an appropriate mechanism is a capability (see for example Needham[2]). However in a network of end systems the mechanism needed is more elaborate. First we need to protect rights sent by telecommunications from being copied or changed in transit. Second we must decide which end systems are to be trusted to handle rights.

The protection of rights on telecommunications lines may be provided by encipherment and associated integrity procedures, as for other types of data. The mutual right to authorise between the basic controller and the CMS therefore needs a mutual means of enciphering and deciphering and the distribution of encipherment keys. To manipulate distributed rights securely, it is likely that each end system will contain trusted software not directly available to application code which ensures that rights are not abused.

4 Examples of use

This section provides several examples to help relate the ideas described to real applications. Section 4.1 describes how some frequently used relationships may be represented in terms of the model. Section 4.2 takes a simple system and describes its secure creation.

4.1 Useful constructions

4.1.1 Connections A user resource, U, is to be permitted to pass messages to a resource S which provides it with a service and to receive replies (results and error indications). The send right allows messages to pass only from the user to the resource used. U therefore has a somewhat restricted form of the authorise right. This enables U to associate an actual right with one specified formal right of S, the one which is used to send replies. U supplies to S the right to send to U to permit return messages. This is illustrated in Fig. 8. This

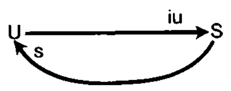


Fig. 8

restricted form of the authorise right is henceforward called 'interactive use' and is marked 'iu' on diagrams.

When the connection is ended U withdraws the send right. It retains the authorise right to enable another connection to be made.

4.1.2 Interpreters A resource, I, may provide the single access path between an end user or resource, U, and a set of subsidiary resources R_1, R_2..., and may provide U with a consistent language which it may use to cause R_1 etc., to do work. This is illustrated in Fig. 9. As in the previous example I is supplied with the right to send to U and each R is supplied with the right to send to I.

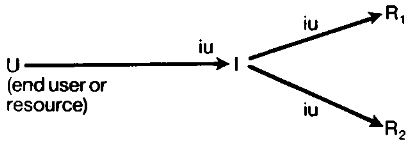


Fig. 9

4.1.3 Service selection services A resource SS may represent a service which gives a user U access to a chosen service. This is illustrated in Fig. 10.

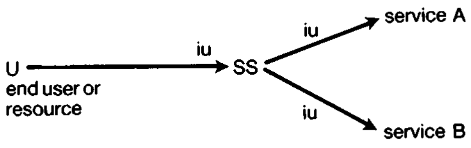


Fig. 10

There are two possible strategies. SS may supply U with the right to 'send to' the chosen service and vice versa. Alternatively, S may relay all messages, enabling it to enhance the language used between itself and the service, and present the language to U.

4.1.4 Authenticators An intermediate resource may act as a guardian which permits a user to access a resource when an authentication has been successfully performed (Fig. 11).

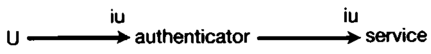


Fig. 11

This may be combined with the previous example so that the intermediate resource authenticates and selects the service required.

4.1.5 Multi-context resources Consider a service which has different users whose access permissions are graded according to their need or trustworthiness. In Fig. 12 'user' may be either P or Q, depending say, on who is sitting at the terminal. He has the right to connect to S as described in previous examples. Authentication is performed in S and, depending on the result, messages are then relayed to SP or SQ. Each of these is controlled by S and has been supplied with the necessary selection of rights (to R and RP in the case of SP and to R and RQ in the case of SQ).

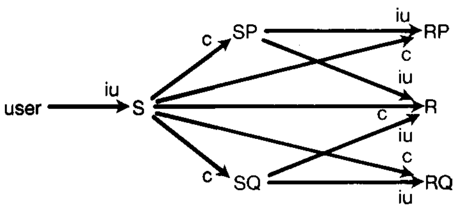


Fig. 12

From the users' points of view they each see a different context of the service S depending on their established identity. In practice, such a multi-context service may be implemented as a complex object rather than in terms of several resources with many rights in common. It may also be convenient to represent a multi-context service as a tree of resources, in which the direction away from the root represents a diminution of the rights.

4.1.6 Private resources Consider a structure of resources which represents a service with different users. A user may be allowed to create resources of his own to which no other user is to be allowed access. He may wish to stop using the service and come back, after an interval in which other users have been active, to access his private resources. This need is catered for by constructing a resource for the user with a set of rights which is identical to that of any other user with the same powers and which, moreover, allows him to create extra resources. Again in practice there is likely to be a more complex 'resource' which allows for individual users' compartments.

4.1.7 Resources as templates Suppose that a resource is a general purpose mechanism which is to operate upon a user's private resource. It may be used by other users also. The user wishes to supply it with the right to access his private resource and be sure that the other users do not constitute a danger. This is similar to the use of a general purpose subroutine. A structure which makes the security needs explicit is shown in Fig. 13.

If user 1 wishes to use the general purpose facility he connects to RG (resource generator), which constructs a separate resource, R1, and supplies

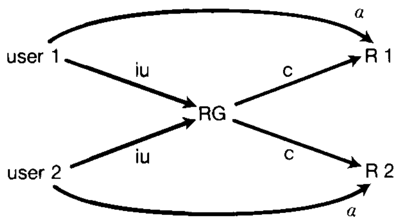


Fig. 13

user 1 with the authorise right to it. User 1 may now supply rights to R1 in any way necessary. The same diagram models a service which offers concurrent use to several users.

4.2 Secure creation of a simple system

Suppose we wish to construct securely the system shown in Fig. 14 with resources in two end systems.

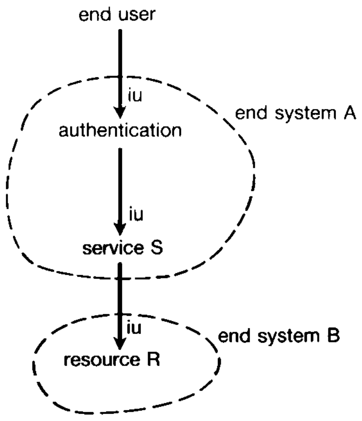


Fig. 14

As described in section 3 the two end systems are created by the management entity, as shown in Fig. 15. For simplicity we may assume that they are the only end systems in the community and that the right to authorise the basic controller at A has been supplied to the basic controller at B. (In Fig. 15 some of the relationships are not shown. The resources at A and B which appear to have no controller are controlled by the management entity).

Now to construct the system shown in Fig. 14 the following is done.

A's basic controller (instructed by the local representative of the management entity) creates a resource 'authentication' and provides it with rights to some of the basic resources, as shown in Fig. 16. The basic controller, since it has

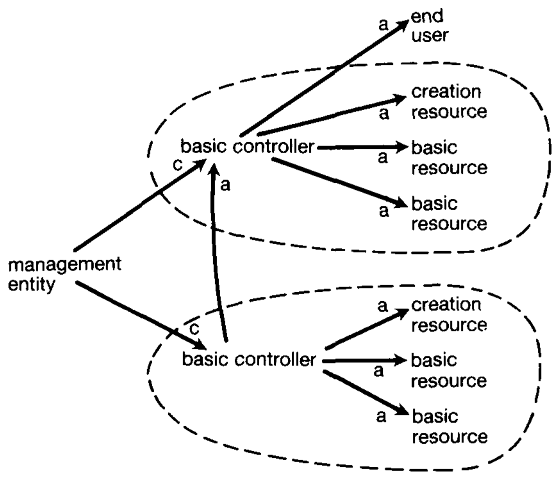

end user
a
creation resource
a
basic controller
basic resource
a
basic resource
a
c
a
management entity
creation resource
a
c
basic controller
basic resource
a
basic resource
a

Fig. 15

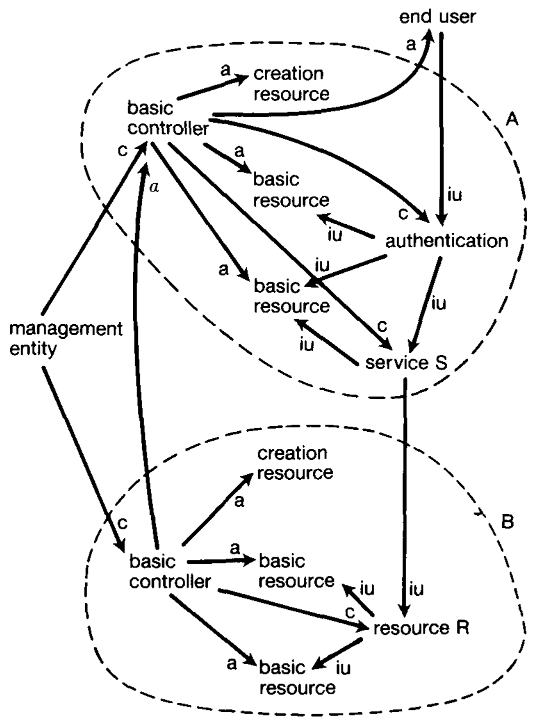

end user
a
iu
A
creation resource
a
basic controller
c
a
a
basic resource
c
authentication
iu
iu
iu
a
basic resource
iu
c
service S
iu
management entity
creation resource
a
B
c
basic controller
a
basic resource
iu
iu
c
resource R
a
basic resource
iu

Fig. 16

the right to authorise the end user, supplies that user with interactive use right to 'authentication'.

A's basic controller creates also the resource 'service S' as shown in Fig. 16 and supplies a right to access it to 'authentication'.

In the same way the resource R is created at end system B. The interactive use right to R is supplied to the basic controller at A, which then supplies it to 'service S'. The rights needed for the application system are now established.

5 Analogies with human systems

5.1 Justification

The objects and operations of section 3 provide a particular way of describing a system and making explicit some of the relationships between its parts. It is pragmatic. It does not derive from an original set of premises, but rather from what appears from experience to be useful. There is an obvious parallel to be drawn between an automated data processing system and one which is provided by people who pass each other information and operate upon it in agreed ways. If the operations and objects of section 3 seem reasonable in human terms it does not prove that they are right or useful. It gives, however, some evidence and may help to provide some insight into what is needed.

5.2 Resources

An analogue of a resource in a human system is a department of a business which provides a particular set of functions to the rest of the business.

The analogue of the code is the set of instructions performed by the staff of the department.

The analogue of the rights of the resource is the set of rights held by the department which defines its relationships to other departments and to people or organisations which are not part of the business.

5.3 Control right

An analogue of the control right in a human based system is the set of powers held by a person or organisation which establishes a department and retains the ability to influence it. In the human department, as for the automated resource, we would expect a procedure to take place when it is created to prepare it for its users. Similarly the operation 'delete' has a parallel. An apparent difference is that a local human manager, as well as the controller, may have the power to close the department down. This may be represented in the automated system by code which refuses all user messages and records the circumstances in variables read by the controller.

The operations of 'activate' and 'suspend' also have analogies in a human based system. In these cases it is even more likely that the ability would be given to the local human manager and the same remarks can be made as for 'delete'. The outside controller should be kept informed and needs the ability to suspend operations in an emergency (and perhaps call in the auditors).

The 'change' operation has an obvious analogue in human terms, together with the need to keep the users informed.

5.4 Authorise right

Section 3 states that the creator of a resource may create the right to authorise the resource. The creator may supply that right to others. This separation of the power to supply rights from other controller powers can be seen also in human systems. Consider for example a credit card. It is used by its owner to authorise another person or organisation to debit the card owner's account and clearly should not permit its owner to affect the ability of the other person or organisation in other ways.

Pursuing the example of the credit card, someone (or some department) has the right to issue a card to its subsequent owner. In doing so he issues a right which enables the receiver to authorise others. Thus the idea of possessing the right to authorise a second entity to authorise yet a third entity can be seen in a human based system.

5.5 Send right

The right to send (to pass a message which will be dealt with by the recipient according to his agreed instructions) is sensibly available in human terms separately from the more powerful operations. The analogy is that of a customer. A customer will usually need a response and the 'interactive use' right described in 4.1.1 therefore has a good analogue.

6 Conclusions

This paper has described a model for a secure distributed system. The model provides the means of creating resources and distributing among them rights to access each other and to provide access to end users. It has described how the operation of the model fits on to a real community of end systems. It has also compared the model to the activities of human organisations which need to communicate securely.

It is intended in a future paper to discuss the representation of rights. The model will be compared with operating system features, for example those of Parker[3], with programming language features, for example those of Brinch Hansen[4], and with other security models, for example those described in Denning[5].

The model as it stands describes the basic rights needed to enable resources to be controlled, rights to be delegated and messages to be sent. It will be useful to define more complex rights. For example, in Fig. 14 if the action of the authenticator were defined then the right of the end user to access service S could be defined. For a given environment a standard authenticator will be useful. This promises to be, therefore, a useful way of enhancing the model. Similarly one may define a 'read only' right to replace a trusted interposed resource which enforces it and a non-transferable right to replace an interposed resource which mediates. One needs to consider also the effect when a right which has been received and passed on or used to supply another right is withdrawn. A system which automatically withdraws consequential rights when the right used to provide them is withdrawn may be built using the basic rights and trusted resources. Such constructions may be defined in terms of the basic concepts and added as mechanisms as necessary.

References

1 International standard ISO/IS 7498. Information processing systems – open systems interconnection – basic reference model.

2 NEEDHAM, R. 'The Cambridge CAP computer and its protection system'. Operating Systems Review Vol. II, No. 5, November 1977.

3 PARKER, T.A. 'Security in a large general-purpose operating system: ICL's approach in VME/2900'. ICL Technical Journal 1982 3(1).

4 BRINCH HANSEN, P. 'The architecture of concurrent programs'. Prentice Hall. First edition 1977.

5 DENNING, D.E.R. 'Cryptography and data security'. Addison-Wesley. First edition 1982.

Mathematical logic in the large practical world

E. Babb
ICL System Strategy Centre, Bracknell, Berkshire

Abstract

The construction of future large information systems will depend increasingly on using rules formalised in logic, rather than on ad hoc algorithms. The beginning of this trend can be seen in database systems, in the older of which the user had to use search algorithms directly in order to answer questions whereas modern systems use query languages based on logic, which automatically invoke these algorithms. The paper discusses the research requirements for this area.

1 Introduction

This paper proposes a long term plan for the evolution of business information systems within ICL to cope with increasing complexity. It responds to the major contemporary problem concerning the confusing number of computer languages and systems. Increasingly, large systems are becoming difficult to maintain in a consistent state, especially where the numbers of users, data and rules become vast enough to be useful. The solution proposed in this paper is to build relatively more of the system in technology independent logic.

Modern database systems are already based loosely on logic as the diagram in Fig. 1 illustrates. For example a query such as ‘Please list the suppliers who do not make parts’ would be mapped to some internal form which could be described by the predicate calculus formula shown. The database interpreter then uses a built in mapping (really a theorem) to change this to some highly optimised search process which in turn drives specialised hardware such as CAFS.

Our long term view on logic is illustrated by Fig. 2. Specifications in logic and algorithms are separated by a sharp break – a logic horizon. The horizon separates technology-independent definitions in logic from technology-dependent algorithms (including PROLOG). Current database systems with their logical query languages are only slightly above this horizon. However, as we attempt to build more complex systems, we believe that relatively more

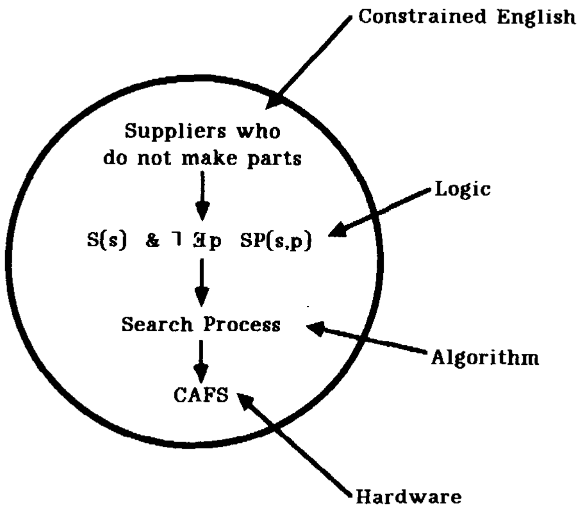


Fig. 1 Logical view of database system

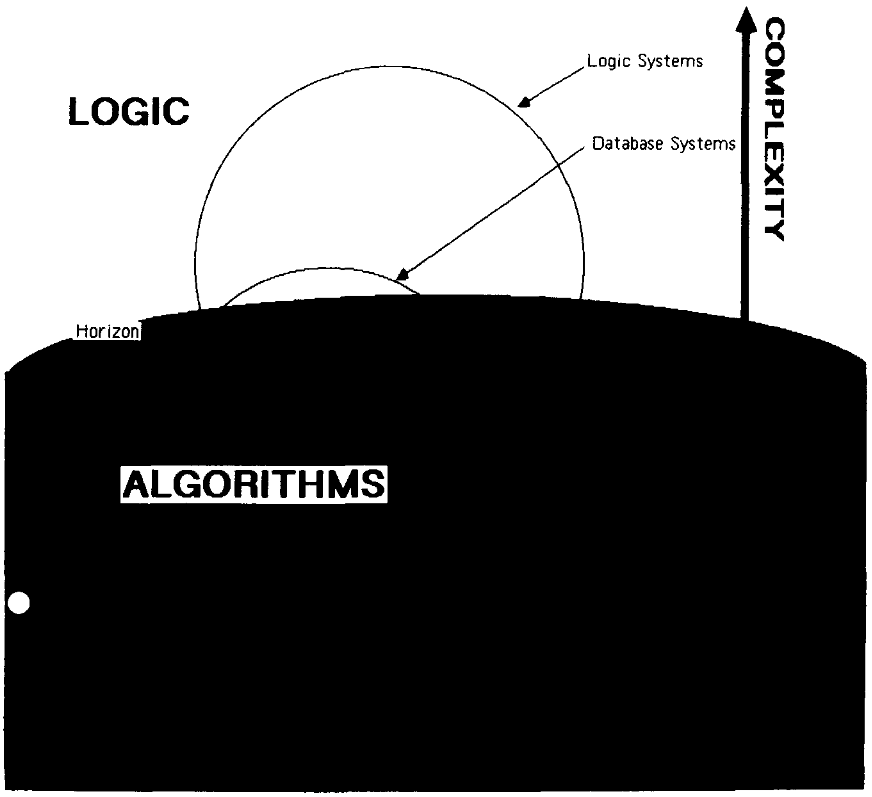


Fig. 2 Logic horizon

of the system will be above and relatively less below. Such new systems are called Logic Systems or Logic Database Systems.

2 Background work

ICL currently sells systems for business modelling using traditional techniques based on data dictionaries, report generators, query languages, Cobol etc. In addition to selling its own systems, ICL also sells a wide range of packages from outside vendors. To run these packages ICL offers a range of business machines such as the 2900 range and the recently introduced series 39 which includes the Content Addressable File Store CAFS as a standard feature. It is important to realise that these are business machines which have different characteristics from high performance single user workstations such as PERQ. The inclusion of CAFS, in business machines, as a standard feature emphasises the importance of searching very large structured files to many ICL customers.

Within ICL there are two main centres of work in logic languages. In Manchester is the Knowledge Engineering Business Centre which supplies PROLOGX. In Bracknell there are two Alvey funded research projects in the Marketing and Technical Strategy (M & TS) division concerned with Logic Database Systems.

ICL is also a partner in the European Centre for Computer Research (ECRC) in Munich. This Centre has a sizeable research interest in logic. It is planned to grow to about 50 people and will research logic programming languages, Knowledge Bases, Expert Systems, Man Machine Interaction and Architectures. Some of the research workers are attached to M & TS mentioned above. ICL also has links with other major academic centres in logic, Edinburgh University and Imperial College. ICL research in this area can also draw on the resources of STC in such places as IDEC and STL.

3 General structure of logic systems

A Logic System has layers between the hardware and the users, as illustrated in Fig. 3. At the top is the dialogue software. This converts for example the familiar notations and graphics of management information systems planners into the logic used in the logic model. The model is a mathematical description of how the system variables being modelled or controlled are expected to behave; it can both answer queries and check the integrity of updates to the rules or data. The interpreter uses these logical descriptions to execute the queries or check the updates.

The different levels of the system are designed by many specialists. The dialogue system might be designed by an industrial artist or a designer using specialised dialogue languages. The logic model of an application might be built by systems analysts, physicists, systems engineers and economists who understand particular applications. The logic interpreter might be built by mathematicians and logicians who provide general theorems to support the

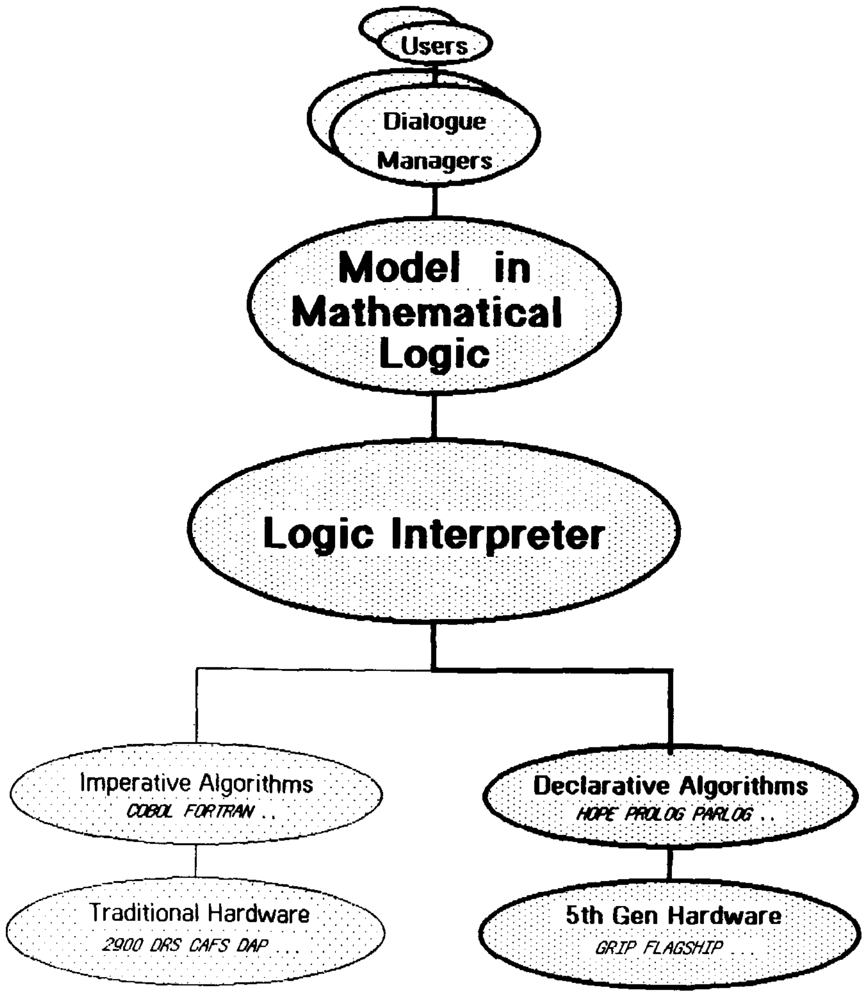


Fig. 3 Logic database system

symbolic transformations required to take a specification to an algorithm. Finally the algorithms and hardware to solve specifications at a more technological level are provided by programmers and engineers.

To users, the Logic System might behave as follows, taking factory control as an example. A systems analyst draws a diagram on a screen describing the flow of raw components, subassemblies and completed assemblies through the factory. The system maps the diagram on to a suitable logic description. Using theorems and standard algorithms the system then simulates, at the analysts command, the flow of finished assemblies, given the flow of raw materials, or vice-versa. The logic description allows much greater flexibility and integrity than a conventional direct mapping of a diagram to algorithms.

4 Logic systems research program

An effective research program has to be related to some application area, otherwise it becomes excessively open ended and divorced from practical considerations. Here, because of their importance, it is suggested that the system should mainly support business applications such as Pay As You Earn (PAYE), planning, forcasting legislation, personnel, purchasing, office information management, retail trade.

4.1 Dialogue manager

Research is needed into dialogue languages and associated software for interfacing users with the logic model. This will be along various lines such as menus or constrained natural language for very casual users, to command-driven interfaces with graphical support involving specialists such as planners, accountants, designers, architects and engineers.

4.2 Logic model

The logic model is illustrated in Fig. 4. At the bottom level 1 are the theorems and algorithms which describe the behaviour of the system variables being

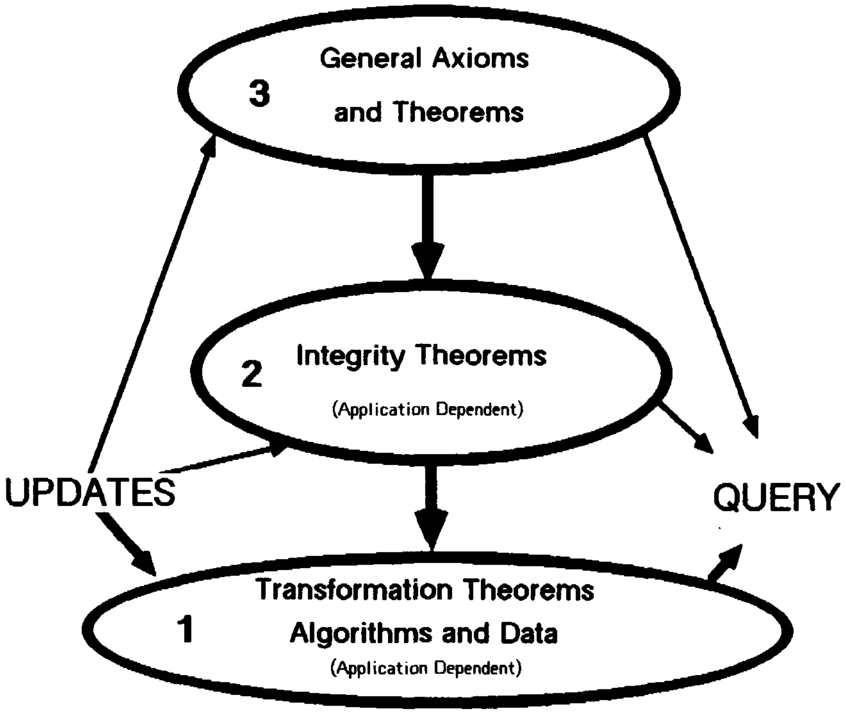
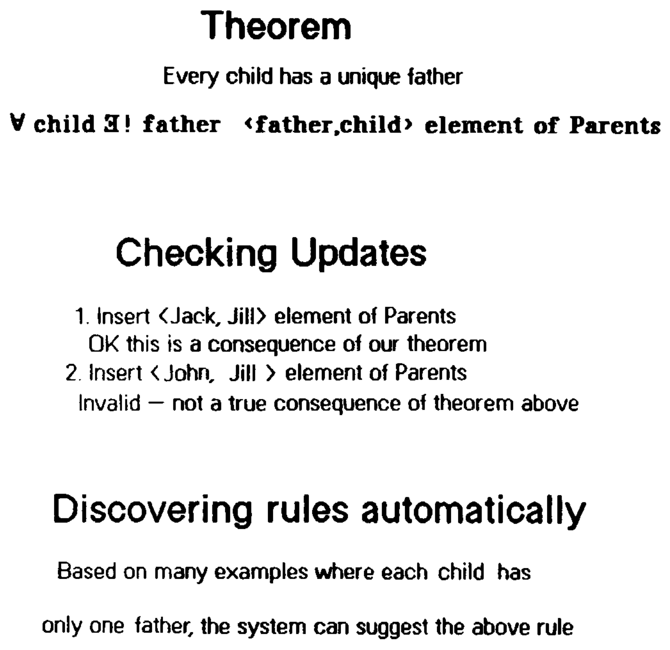


Fig. 4 Logic model

modelled. For example how an employee's wage depends on the date and on age, seniority etc. The next level 2 contains the more general integrity rules which can describe a whole range of applications. For example, in Fig. 4a is a theorem stating that a child can have only one father: this theorem can then be used to check updates to a set of father/child pairs. At level 3 are general mathematical properties of algorithms (or theorems): for example, the associative properties of certain arithmetic operators. Each level must always be a logical consequence of the level above.

Queries are usually applied to level 1 to find out how our model is behaving. Updates also are usually applied to level 1 but are only allowed if they lead to a new state that is a valid consequence of the rules in levels 2 and 3.

Research is required into the general problems of building systems in logic, in particular into the discovery of useful transformation and integrity-checking axioms or theorems. Research is further required into automatic methods of discovering these rules, for example discovering the theorem of Fig. 4a or the rules in areas where the precise values of numerical parameters change with time – such as the tolerances in the dimensions of an object. Alternatively it might be the actual rules of logic itself that alter with time: for example, suppliers and parts may at first exist independently and then later exist only as a pairs in a supplier-part catalogue.



Theorem

Every child has a unique father

$\forall$ **child** $\exists!$ **father ‹father,child› element of Parents**

Checking Updates

1. Insert ‹Jack, Jill› element of Parents
   OK this is a consequence of our theorem
2. Insert ‹ John, Jill › element of Parents
   Invalid — not a true consequence of theorem above

Discovering rules automatically

Based on many examples where each child has

only one father, the system can suggest the above rule

Fig. 4a Examples of logic model

4.3 Logic interpreter

The general function of a logic interpreter is to solve problems expressed in logic. Fig. 5 shows a logic universe where points represent formulas. Standard algorithms can be represented by single predicates, for example a search algorithm or an algorithm for finding the square root of a number. Problems are solved by transforming a point in this space to a set of standard solution algorithms, for example transforming a complex database query in the upper region to a sequence of standard algorithmic solutions, such as a search algorithm, in the lower region.

The logic interpreter is the key to the whole system. It must provide the following features, described in more detail in the next paragraph.

exactness so that expressions can be easily transformed by a theorem prover

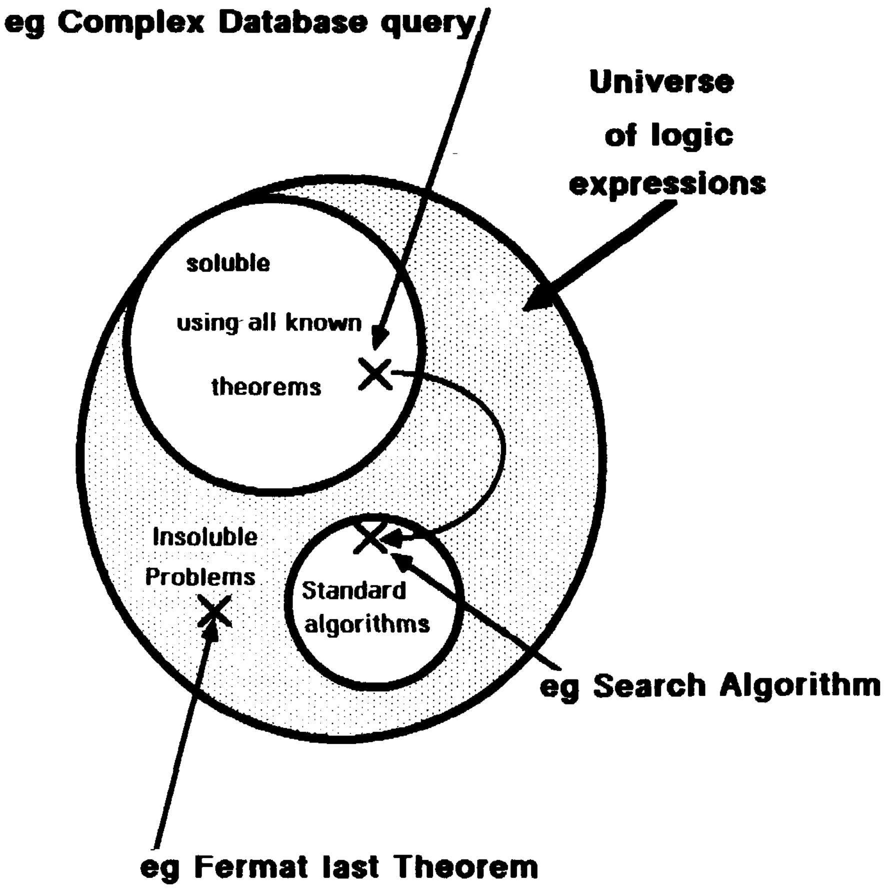

eg Complex Database query
Universe of logic expressions
soluble using all known theorems
Insoluble Problems
Standard algorithms
eg Search Algorithm
eg Fermat last Theorem

Fig. 5 Logic universe

explanation so that users can modify queries and updates easily and be confident about the system's behaviour
reversibility to improve the power of the language
theorems to equate equivalent expressions
computability to trap infinite processes
algorithms, architectures and *networks* to supply the performance needed to solve the problems submitted in a cost-effective manner.

4.3.1 Exactness This means that the semantics of our logic is based on predicate calculus. Logic programming languages such as PROLOG have many impure features which make them difficult to transform using a theorem prover. In reality they are powerful relational programming languages and would in the proposed system be used for defining algorithms.

4.3.2 Explanation When the system answers a query or executes an update to the model an explanation is often required as to what the system has done, even when this would appear to be intuitively obvious. This is especially true if the system is unable to answer a query and the user wants to know what extra information the system needs in order to solve his problem.

4.3.3 Reversibility Datebase systems are usually reversible. The user can for example ask either for the parts made by a particular supplier or for the suppliers of a particular part. Alternatively a system might be described by a set of simultaneous equations – a matrix transform from a vector x to a vector y. The specification needs to give both the forward calculation and where computable the reverse calculation based on the inverse matrix. Such reversibility is often used by designers who want the output of some system to be fixed and then want the input condition corresponding to this output.

4.3.4 Theorems To solve any problem in mathematics theorems are used. These map a specification on to either a set of axioms or on to a set of sub-problems, each of which can be solved by means of a standard algorithm. For example, one method of solving a differential equation is to transform it by hand into a form for which standard solutions are known to exist.

Many new kinds of theorem are required. Examples are termination theorems for trapping infinite loops; theorems for examining the intrinsic computability of a process; optimisation theorems for transforming a slow process into a fast one; theorems for reasoning about time.

4.3.5 Computability It is essential that the logic interpreter should be able to discover when it cannot solve a particular problem, rather than go unhelpfully into an infinite loop. In practice this involves trapping those conditions that might cause an algorithm or theorem to loop. This can be done by including special guards in the definition based on further theorems.

4.3.6 Algorithms Discovering new algorithms and novel machine architectures is a highly creative process. For example, Newton's discovery of the

Binomial Theorem allowed the transformation of previously uncomputable statements into series for which values could then be approximated. Similar invention went into the discovery of the original Von Neumann computer architecture.

There are two separate classes of algorithm which are of interest: algorithms for solving conventional problems involving numeric or symbolic data and meta algorithms which transform logic statements and are used directly in the logic interpreter. These algorithms are currently written in traditional languages but the trend now is towards languages which can explicitly control parallelism. Studies are required into the various languages on offer and whether they can be united into a single language such as a combined logic/functional programming language.

Very often the system will find that it cannot solve a problem because no algorithm currently exists for its solution. For example, imagine the specification of a square root where the algorithm for squaring a number exists but not its inverse. In these circumstances basic research needs to be performed into methods of discovering such inverse algorithms automatically.

4.3.7 Architectures and networks Special hardware architectures may be needed for theorem transformation. This is an area urgently in need of research.

An important feature of logic is its potential as a common intermachine communication language. Rather than complex low level protocols which differ from machine to machine it should be possible to make logic the main form of intermachine communication. Thus problems would be defined in logic and solutions returned as obviously executable logic expressions.

5 Conclusion

The paper has merely expressed and clarified the general trend of an ICL policy already in being, that has the direction of the views given here. The company is encouraging research in many of the areas described in universities and other institutions, by collaborations supported by the ICL Universities Research Council or by the Alvey and Esprit programmes. In this way the skills of talented individuals outside the company can be used, to the benefit of both sides.

The ICL DRS300 management graphics system

R.J. Bunyan
ICL Office Systems, Reading, Berkshire

Abstract

The paper describes the principal graphical aids to managerial and office work provided by the ICL DRS300, the latest addition to the DRS family of networked products.

1. An overview of management graphics.

Graphics in the office has really appeared only in the last three years. Graphics was previously the preserve of CAD/CAM and specialised, vertical market applications, the computer power necessary to drive these systems making them too expensive for personal business graphics use. More recently specialised graphics processors, for example the NEC 7220 and the Motorola 6854, have been produced in silicon at relatively low cost; and this, coupled with the ever-falling price of colour bit-mapped screens and with the availability of cheaper, faster processors has made personal management graphics a commercial reality.

This commercial reality began with the Apple Mackintosh, which was the first sucessful personal computer to reach the market with high resolution graphics, albeit monochrome. Previous systems had been costly and slow. Now there are a number of personal computers with fast, high resolution graphics; the DRS300 Management System is unique in being a true multi-user system with high resolution colour graphics and 'on board' networking – together with all the benefits of DRS300 and the DRS family.

Management graphics – graphics in the office – performs four major roles:

Graphing. First and probably the most obvious is the ability to present statistical data in graphical form, for example as pie charts, histograms etc.

Presentation. Second is to provide a medium for presentations and for presentation preparation. This can be achieved by projecting the video Red-Green-Blue signal on to a large TV screen or by creating 35mm transparencies using such devices as the Polaroid Pallette, which takes the composite signals from a video and photographs them on to a 35mm transparency.

Overhead transparencies can be created and edited on the screen and then printed on to acetate foils, using for example an ink jet printer.

Applications. A number of office applications other than the obvious graphing benefit from graphics. For example: inclusion of drawings or pictures (e.g. logos) into word-processor documents; design of forms; production and subsequent transmission of maps and such things as pipe or cable layouts. Also, horizontal applications such as spreadsheets and databases can exploit graphics facilities to great effect.

Man-machine interface. By using graphics a far more user-friendly interface can be presented to the end user than that provided by the more traditional character screens. Graphics facilities allow pictures ('icons') to be used instead of text and items can be selected by 'pointing' at them, using the mouse. The ability to redraw an image quickly permits text or images to be 'cut' from an area and 'pasted' elsewhere.

2. The DRS management graphics system.

The system comprises hardware and software components; in the following accounts it is assumed that the reader is already familiar with the general DRS300 System hardware.

2.1 The hardware

The graphics hardware components are

the Submodule 5 (SM5)	– the graphics controller card
the Model 307 monitor	– consisting of bit mapped, memory mapped video with mouse and keyboard
various output devices.	

The SM5 provides a bit mapped, memory mapped display which can be displayed on either a colour or a monochrome display; and apart from its graphic functionality provides a text plane for high speed manipulation of character displays. It provides a colour palette enabling any 8 colours to be displayed from a selection of 64.

There are three graphics planes and one text plane. For the graphics plane, each pixel on the screen is defined by one bit from the same position on each plane; the three bits are mapped by the colour palette to provide four intensity levels for each of the colours red, green and blue and hence up to eight colours can be selected from the 64 available in the palette RAM. This is illustrated in Fig. 1.

Each graphics plane is 32K bytes in size, giving a total of 96K bytes for the

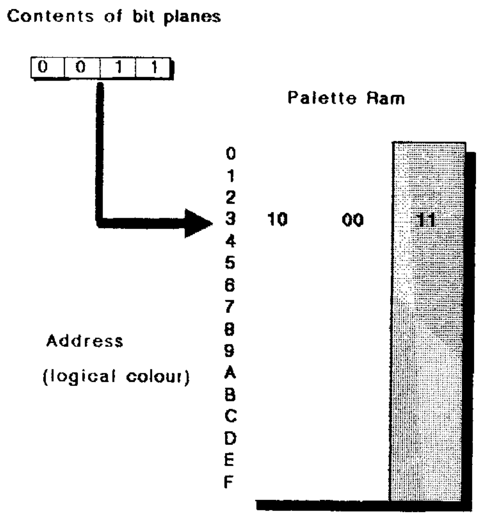


Fig. 1 The Hardware Colour Palette

graphics submodule. However, since the submodule is restricted to 32K bytes of the 80286 processor's address space a page register is used to provide a bank select for the memory on the submodule, there being a bank for each of the graphics planes.

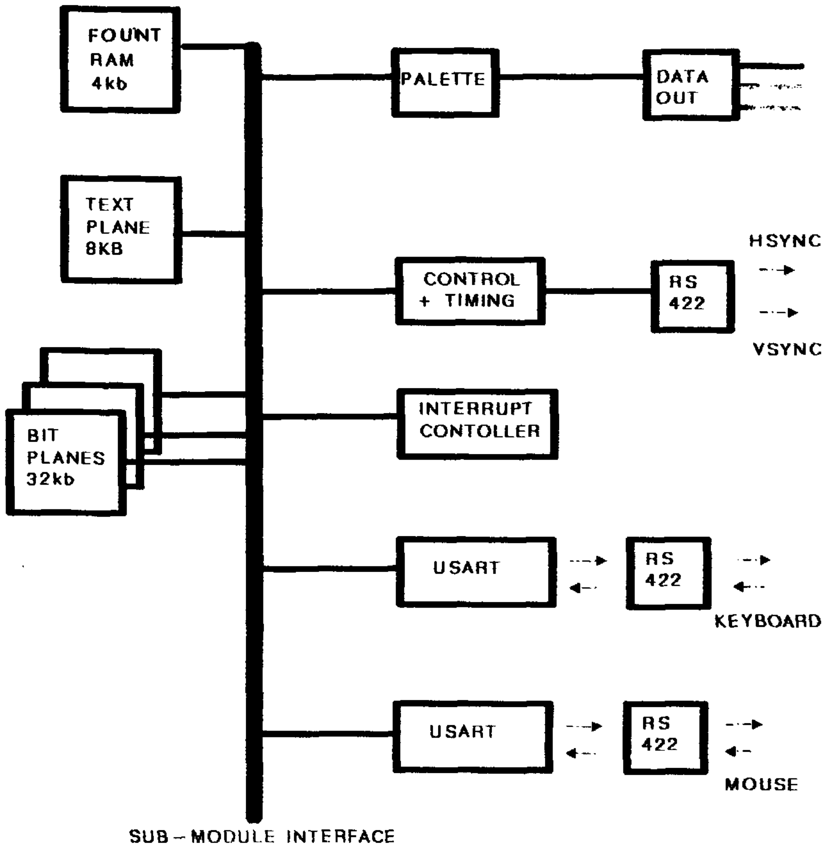
Data transfers to or from the video memory take place on complete byte or word boundaries in the selected plane. The SM5 provides a 'multi-plane write' facility, enabling a memory write operation to take place on all three graphics planes simultaneously with different data; hence a particular colour can be written to a row of 8 or 16 pixels in one memory cycle.

The SM5 uses the NEC 7220 Graphics Display Controller chip (GDC); this provides the following functions:

- generation of the basic video raster timing, including synchronisation
- calculation of pixel memory address for figure drawing
- figure drawing operations
- area filling

Text operations are provided by a dedicated text plane, available either in lieu of or superimposed on the graphics plane. As with the graphics plane, 8 colours from a palette of 64 can be displayed. However, additional attribute bits can be selected, permitting 8 foreground and 8 background colours to be selected. Character founts are held in a RAM and can be redefined by software.

Figure 2 gives a block diagram of the SM5.



SUB – MODULE INTERFACE

Fig. 2 Sub-Module 5 Block Diagram

USART:	Universal Synchronous/Asynchronous Receiver/Transmitter
HYSYNC:	Horizontal Synchronization
VSYNC:	Vertical Synchronization

The Model 307 Monitor is a memory mapped, bit mapped colour monitor with a resolution of 640 x 400 pixels. The screen is refreshed at a rate of 60 hz. It is connected to the SM5 by a cable that can be up to 10 metres in length. As well as the keyboard an optical mouse is provided, using the same cable to connect to the SM5.

2.2 The software: GEM.

The system employs the Digital Research Inc. Graphics Environment Manager (GEM), running with the Concurrent DOS86 operating system. GEM has the same origins as the ICL PERQ, and indeed as the Apple Mackintosh, an open study on man-machine interfaces undertaken by Xerox Corporation at their Palo Alto Research Centre in the late 1970s and known as the Xerox STAR Project. The objective of STAR was to demystify the system by moving away from jargon and enabling the user to communicate with the computer in terms that he would use in normal office practice, that is, in terms of 'metaphors' – commonly referred to as Windows, Icons, Mouse and Pointers – WIMP! The metaphors evolved were –

- screen windows, for paper
- office objects, e.g. waste bins, filing cabinets; these become icons or small pictures
- actions on these objects, invoked by pointing at them with a mouse so that an office process becomes a computer process.

These design goals imply a set of system requirements in terms of high resolution graphics and high speed graphical object movement on the screen. The ICL PERQ was one of the first machines to use these facilities.

GEM is the first product to provide a truly portable interface for graphics applications, together with a friendly 'WIMP' interface. In this it contrasts strongly with many other products on the market, where application writers are required to write to the video memory map directly, thereby foresaking any portability and perpetuating an archaic architecture.

GEM's four main functions are to provide the following:

- man/machine interface – a WIMP style interface, similar to that on the Apple Mackintosh
- applications interface – a portable interface for graphics applications tools
- a number of software tools and libraries for writing applications to work with the GEM man-machine interface
- graphics primitives – for implementing graphics applications with reduced programming effort.

It has two main components, GEM Virtual Device Interface (GEMVDI) and Applications Environment Services (AES); these are described below.

2.21 Virtual device interface, GEMVDI.

The role of this is to provide the primitives for graphics applications; drivers for specific devices translate the standard GEMVDI calls into the particular characteristics of each device, thus providing device independence. It has two components, the Graphics Device Operating System (GDOS) and the Device Drivers.

The GDOS contains the basic device-independent graphics functions that can be invoked by an application. It performs co-ordinate scaling to translate the application's 'virtual' co-ordinates into the corresponding values for the particular graphics device. Alternatively it permits the application to write to raster space, in which case no co-ordinate transformation occurs.

There is a Device Driver for each graphics device in a system; each communicates directly with its associated device and translates the GEMVDI primitives into the inherent capabilities of that device. In some cases the driver emulates facilities not provided by the hardware.

2.22 Applications environment services (AES).

This component provides a comprehensive set of library routines to simplify the task of writing GEM applications: for example, the GEM DESKTOP application described in the next section uses only AES routines and does not address GEMVDI directly. AES provides also a number of services for running graphics applications, such as –

- handling drop-down menus via the MENU BUFFER
- tools to build a graphics resource file (Dialogue boxes, Alerts, Icons etc.) together with interfaces to access these at run time
- event monitoring, handled by the Screen Manager which responds to mouse movements or button clicks occurring outside the currently active window. The Screen Manager sends the results of such mouse movements to the active window
- scheduling, performed by the Dispatcher which schedules CPU time between the various active processes and ensures that no one process dominates to the exclusion of the others. The Dispatcher apportions CPU time between the primary application, background processes and Screen Manager by assigning each task to either the Ready List or the Not Ready List. The latter consists of those processes that are awaiting an event such as a mouse movement, keystroke or message; those on the Ready List are not waiting for an event and are scheduled on a 'round robin' basis.

The AES subroutine libraries provide routines for use by applications writers for a wide variety of tasks such as windowing, monitoring mouse movements, displaying system or error messages; also for Icon selection, window sizing and scrolling.

2.23 GEM applications and accessories.

GEM applications are generally written in C language and bindings are available to enable applications to exploit the AES interface. One such application is GEM DESKTOP, which provides the distinctive man-machine interface. Objects in the system are represented by icons – small pictures of the objects they represent, selected by pointing at them with the mouse. The DOS hierarchic filestore is represented by Folders (representing Directories) which are opened by selection with the mouse to reveal their contents displayed as icons.

Desk Accessories are small applications that occupy a window that is less than the whole screen and run concurrently with other applications or accessories; up to 6 may run concurrently. They consist of processes such as a calculator, clock, spooler etc. The system provides additional accessories such as the colour palette which enables the user to 'mix' the colours displayed on the screen. Colour changes may be saved as a palette file for subsequent recall, so that a user may, for example, save a colour palette that matches the ink colour of his ink jet printer of graph plotter. By use of 'dither' – adjacent pixels of different colour giving the appearance of a third colour – a practically infinite number of colours can be displayed on the screen.

A number of GEM applications have been produced covering a wide range of activities, such as databases, low cost CAD, Knowledge Engineering, and desktop publishing. GEM applications currently issued by ICL include –

GEM WRITE	– a wordprocessor that enables graphics to be included in a document
GEM DRAWN	– a drawing package using vector graphics
GEM PAINT	– a raster-based application for pixel-level painting and editing
GEM GRAPH	– a business graphics package that permits data to be imported from a spreadsheet or keyed in, and displayed in a variety of graphical forms
GEM WORDCHART	– an application for preparing presentation material

Additionally, a series of utilities have been produced which exploit the GEM man-machine interface and provide for disc formatting, copying, archiving etc.

Because application writers use the AES and because of the standards previously set, all GEM applications have the same intuitive man-machine interface.

Non-graphics applications. It was mentioned above that the SM5 supports a text plane; the system exploits this and can run up to 4 virtual consoles or windows, using the text plane concurrently with GEM on the graphics plane. The user might use one of these virtual consoles to communicate with a mainframe and the other perhaps to run a spreadsheet such as Lotus 1-2-3. It then becomes possible to import the spreadsheet into GEM, the user switching between GEM and the text-based virtual consoles by a simple key depression.

The system may also be networked to other DRS300 systems, using any of the standard DRS300 transports. Access to remote resources is made simple by displaying those that are available as icons along the bottom of the GEM screen; the user points to the appropriate icon and clicks twice on the mouse button, whereupon the device is 'opened' and displayed in standard DESKTOP format. Similarly data can be transmitted over the network by simply clicking the source and 'dragging' it to the target icon.

The general structure of the graphics is shown in Fig. 3. All the figures in the paper were produced on a DRS300 Graphics Workstation system.

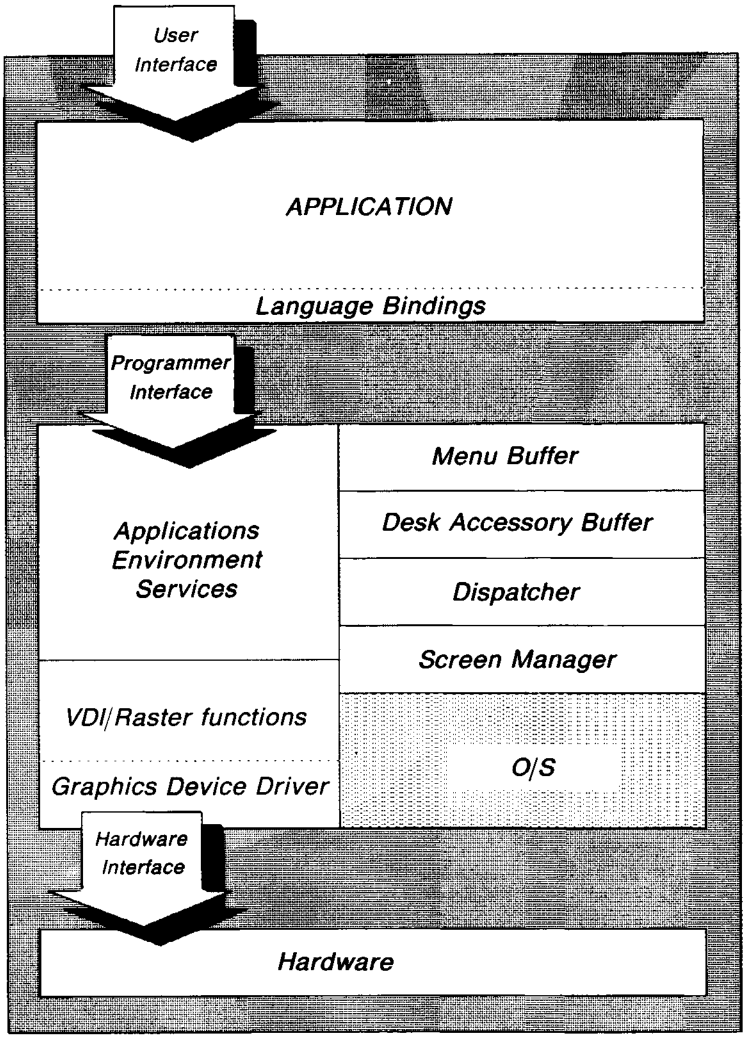


Fig. 3 Structure of DRS300 Graphics System

Performance of OSLAN local area network

A. Maynard-Smith
ICL Network Systems, Stevenage, Hertfordshire

Abstract

This paper describes the results of a simulation of the performance of the OSLAN local area network, and relates them to theoretical predictions, and results from practical measurements. Simulation has been used because of the difficulties of mathematical analysis of the system on the one hand, and the large cost of equipment and effort required for practical measurement on the other.

The main direction of the simulation work has been in providing practical guidance on the performance limits of OSLAN systems in actual use. In general this has shown that bottlenecks occur elsewhere in the system before the OSLAN becomes a limiting factor.

1 Introduction

ICL's strategic direction for the interconnection of computers and workstations within a building or site is the OSLAN local area network. This is a network system conforming to international Open Systems Standards[1,2,] which are in turn based on the original Ethernet system[3].

The OSLAN local area network is a CSMA/CD (Carrier Sense, Multiple Access with Collision Detect) contention bus, running at a nominal speed of 10 Mbits/sec. Such a network may have many hundreds of devices attached to it, each able in principle to communicate with all the others. In a practical situation, of course, the pattern of intercommunication will be dictated by the applications and services each device requires to access or participate in.

It is important for practical network planning to understand how an OSLAN network behaves when a large traffic load is placed upon it, what throughput and response times are experienced, and where the performance limits occur. Theoretical studies of CSMA/CD performance[4,5], and comparisons of the performance characteristics of CSMA/CD with other local area network technologies[6] have been published, but little is available which answers the questions above in a form suitable for practical application. This paper attempts to provide such information.

The prime method used was simulation since the CSMA/CD protocol is difficult to analyse theoretically, and actual measurement over the full range of system parameters would require a very large amount of equipment to be available.

2 The OSLAN network

An illustrative OSLAN network is shown in Fig. 1. For the purposes of this paper the following main points should be noted:

The main network is a branching tree made up of a number of Segments joined by Repeaters, the simplest case being a single Segment. The conditions controlling the maximum size of network are somewhat complex[1], but are based on setting a limit to the maximum signal propagation time between any two points on the network.

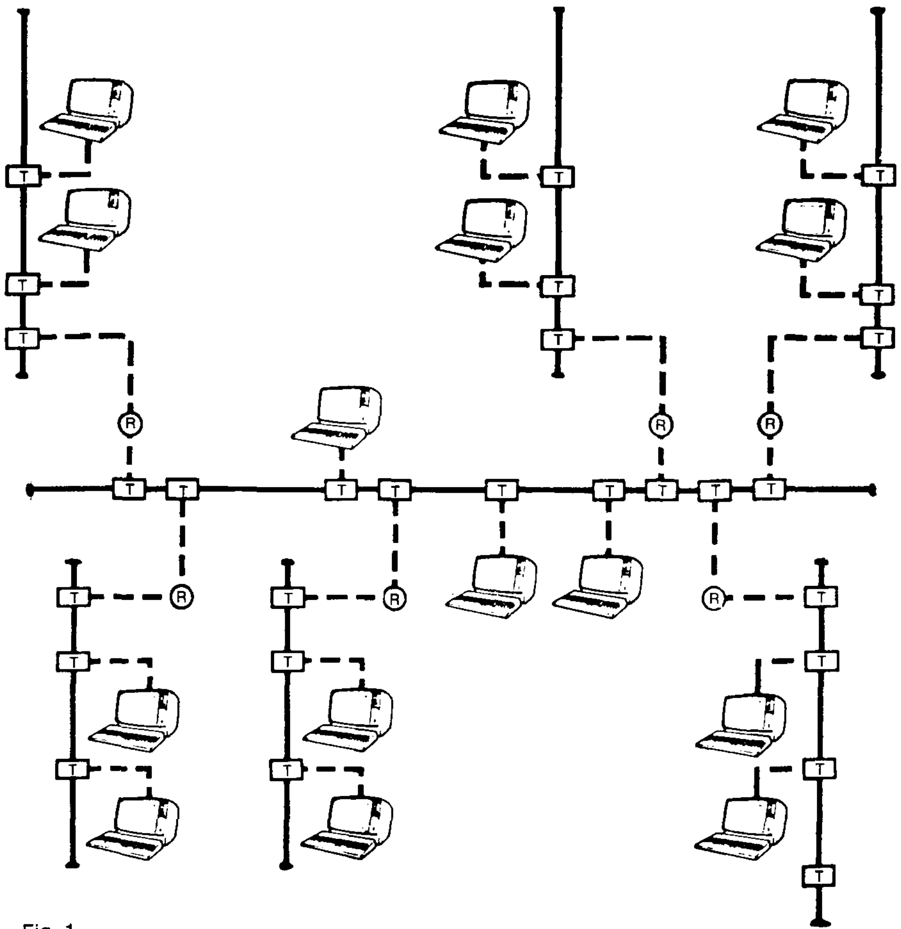


Fig. 1

The network is operated on the contention principle with any station being free to transmit, so long as no one else is transmitting already, and so long as transmission ceases (according to defined timing rules) if two stations try to start talking at once. In the case of such a collision a randomising process governs the delay before retransmission attempts, to avoid simply repeating the collision.

There are maximum and minimum limits placed on the size of frame which may be transmitted, to ensure firstly that no station holds on to the network for too long having gained access, and secondly that a potential collision is always recognised as such before the frame transmission is complete.

It should be noted that this description of OSLAN only covers Layers 1 and 2 of the ISO Reference Model, and that in a practical system there are various higher level protocols which affect the operation of the system as a whole.

3 The simulation method

The results were obtained using the internal ICL program ESIM, which is a discrete event simulator, written in PASCAL, and currently running on an ICL 2900 under VME. It is basically a general purpose system, capable of modelling many types of communications network, but with specific capability built in for simulating the CSMA/CD protocol of OSLAN.

ESIM provides a method of defining a set of message generators, queues, activities which represent the processing elements of a system, message sinks, and as a special case an OSLAN network. The simulated size and arrival rate of messages at each generator can be controlled to place a specific loading on the simulated OSLAN, or on any other resource being examined.

Having defined the network to be simulated the program is activated to track all the significant events occurring, and change the state of the model appropriately on each event. Some of the events which are recognised by the model are:

- Completion of a specified processing delay.
- Arrival of a message in a queue.
- Completion of the transmission of a message on the simulated OSLAN.
- Completion of the propagation of a signal across the OSLAN.
- Completion of the 'Backoff' delay after a collision.

Whenever such an event occurs an activity, defined in setting up the model in the first place, is activated to take the relevant action. Such actions include:

- Starting transmission of a message on the OSLAN.
- Scheduling a message generation event at some future time.
- Placing a message on a queue, to be serviced by another activity.

– Initiating a delay representing simulated processing or line transmission time.

During the simulation a number of statistics on data throughput, message transit time, queue size, etc., are collected, and may be displayed on demand.

4 Conventions and system parameters

The results below are given in terms of Transfer Time for various values of OSLAN Utilisation and other parameters. The usage of these terms is described here.

4.1 *Transfer time*

The one-way time from the arrival (i.e. generation) of a message until its receipt at the far end. This includes any queuing at the transmitting station, deferring to passing traffic on the OSLAN, backoff and retry after collisions, and actual transmission time. It does not include any allowance for processing in either station, nor impact of higher level protocols.

4.2 *Utilisation*

The fraction of the 10 Mbit/sec bandwidth occupied by all bits of successfully transmitted messages. This includes all header, framing and synchronising bits, and the mandatory minimum inter-frame gap, but excludes other idle time and all transfers which result in collisions.

The Utilisation is given by:

$$\text{Utilisation} = \frac{\text{Mean Arrival Rate} \times \text{Mean Gross Size (octets)}}{1\,250\,000}$$

summed across all the stations on the network.

4.3 *Frame size*

The gross size of a frame for the current phase of OSLAN protocols may be calculated as follows (future developments will increase the size of the Network Layer Header in particular):

Transport User Data (including higher level headers)	L(T)
Transport Layer Header	5 octets
Network Layer Header	1 octet
Link Level Header and Frame Check	21 octets
Padding to make total of above up to 64 octets (if reqd.)	
Synchronising Bits	8 octets
Inter-Block Gap (equivalent)	12 octets

Or combining these figures:

Gross size = max[L(T) + 27, 64] + 20 octets

It should be noted that the size of importance is the size of the frame actually transmitted on the network, not necessarily the size of a message presented at some higher level, since fragmentation of large messages is to be expected.

Note also that this calculation gives a minimum frame size of 84 octets, which in particular means that all Acknowledgement messages generated by higher layer protocols are of this size.

4.4 Frame size distribution

Frame sizes are quoted gross, as described above. Results are given for constant sizes, and for random sizes following a negative exponential distribution. A random distribution is however only meaningful in the middle of the size range, since near the defined maximum and minimum values there is a sharp cut off. In the extreme to get a mean size of 1500 octets, the size must be constant at 1500.

4.5 Propagation time

One significant factor in the results is the signal propagation time from one end of the OSLAN cable to the other. Some typical values are:

2 microseconds	A minimum size system with all cable lengths near to zero.
5 microseconds	A full length, single segment, system.
25 microseconds	A maximum multi-segment system, with full complement of repeaters, maximum cable lengths, etc.

In fact for simplicity ESIM assumes that the OSLAN is a single straight line, and ignores the possibility of branching topologies using repeaters. Such topologies have the effect of increasing the mean propagation time between two points without increasing the maximum value. The ESIM results may be slightly optimistic in such an extreme case, but this is unlikely to be of major significance.

4.6 Concurrency

The model used in obtaining most results assumes that each station has a Poisson arrival process generating messages, and that all arrivals are independent. This approximates to a station with a large number of independent terminals or multiplexed data streams making up the traffic load. Stations with a small number of streams are more complex since the effects of message fragmentation, bursty arrivals, interaction with higher level protocols, etc., become significant.

Some results are given however for stations having a single data stream. This is modelled as a process with a random (negative exponential) inter-arrival time, but which only starts running the inter-arrival timer after the successful transmission of the previous message.

5 Validation and limitations

One of the important questions with any simulation process is ensuring that the results do actually reflect reality. Of necessity some simplifying assumptions have to be made in constructing the model, and the simulator is a piece of software just as prone to bugs as any other. Therefore a validation of the model against independent criteria is essential before the results can be trusted.

5.1 Validation

ESIM has been tested in a number of ways, the following areas in particular being examined:

- Consistency of results in simple cases, e.g. the time taken to transfer a message with no other activity present, correct generation rate of messages by random processes, etc.
- Correct results from models of queuing systems with known analytical solutions. This includes simple queues, multi-server queues, limited population cases, etc.
- Manual examination of the sequences of events generated by the CSMA/CD propagation and collision algorithms, including the case of three-way collisions.
- Comparison of the results of simulation with measurements on real systems[7]. This study measured the maximum throughput obtainable on an Ethernet system with varying packet sizes and numbers of stations. With 64 stations active, and over a range of packet sizes giving saturation loads from 54% to 97%, ESIM gave results consistently within 2% of the actual measurements.

5.2 Limitations

Two further comparisons with actual measurements expose a limitation of the simulation technique used. These comparisons were with the observations[7] for a system containing only 5 stations, and with some test measurements on ICL products, also having only a small number of stations active. In both these cases the ESIM results are very variable, bracketing the observed measurements, but highly sensitive to small changes in the parameters assumed.

The common factor in both these cases was that a small number of stations were running flat out to generate the load on the network, transmitting as fast as they could. In these circumstances instead of having random arrivals the load generators tend to have constant repetition rates, and they get into lockstep, thus greatly reducing collisions. It takes only a small extra element of random delay between messages to destroy this pattern and change the system behaviour dramatically.

Because the behaviour of this type of load generator is not known in detail it is not possible to simulate it with confidence. Fortunately however this is typical of a test system set up to generate as much traffic as possible, not of a practical computing system with many other tasks to perform. This observation does however underline the importance of the assumptions about random arrivals, and warns that we should beware of circumstances in real systems which may make this assumption invalid.

5.3 Accuracy

By its nature a simulation process will give results which are subject to statistical error, even if no systematic error is present. By examining repeated runs with identical parameters the statistical error may be estimated. For the figures presented in this paper the statistical error in the values of Utilisation may be expected to range from 1% to 2%.

5.4 Conclusions on applicability

The OSLAN simulation described in this paper is an appropriate method for systems with reasonably large numbers of stations, and/or with reasonably random arrival patterns. This is expected to cover most cases of practical interest.

For systems with small numbers of stations and approximately deterministic arrival patterns the simulation is unproven, but in this case the actual behaviour is very sensitive to small changes in parameters, which makes any detailed prediction dangerous. This is not expected to be a common case in practice.

6 Results of simulation

This section presents the results of simulations under a variety of conditions. Except where indicated otherwise all are for systems having 100 active stations, with traffic equally distributed between them, and random arrivals according to a Poisson arrival process.

6.1 Transfer time results

Fig. 2 shows results for the Transfer Time of random sized frames, plotted against Utilisation, at various values of mean gross size. Fig. 3 shows the same results but with the transfer time normalised, with respect to the mean Service Time (i.e. the actual transmission time). The curves show a typical

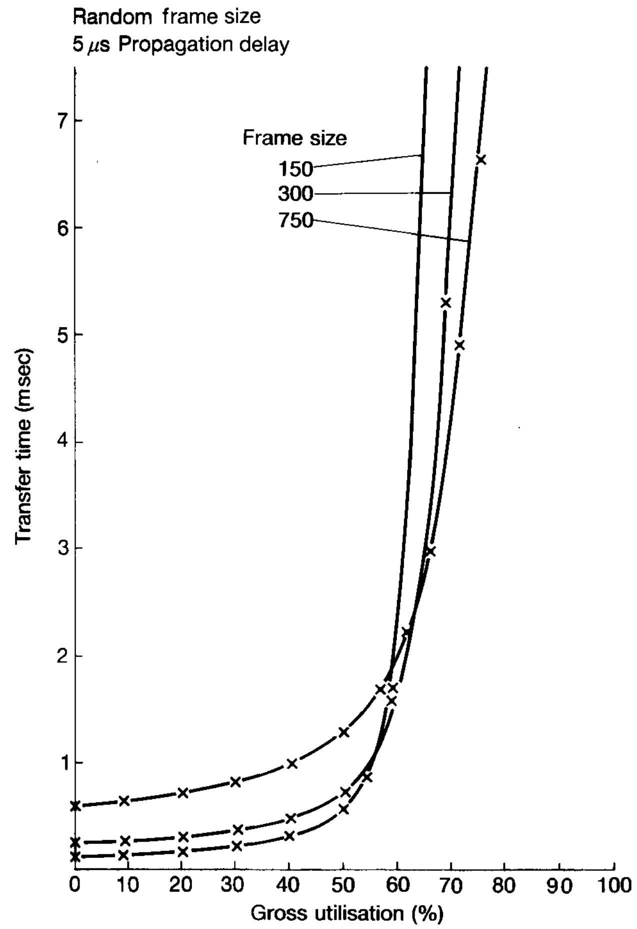


Fig. 2

queuing situation, with the larger frame sizes having a better normalised response time, and a higher limiting value of achievable Utilisation.

Fig. 4 shows similar results for constant sized frames, which have a very similar shape. As pointed out above, frames of 85 or 1500 octets mean size must be constant size, so extra curves are included in these diagrams. In common with other queuing systems constant sized frames have a better Transfer Time than random ones.

Fig. 5 shows the effect of propagation delay (i.e. of the physical size of the network) on minimum size frames. For larger frames the propagation delay makes increasingly little difference. The propagation delay affects the Transfer Time because a longer delay gives a larger window in which collisions can occur, and more collisions cause more delays.

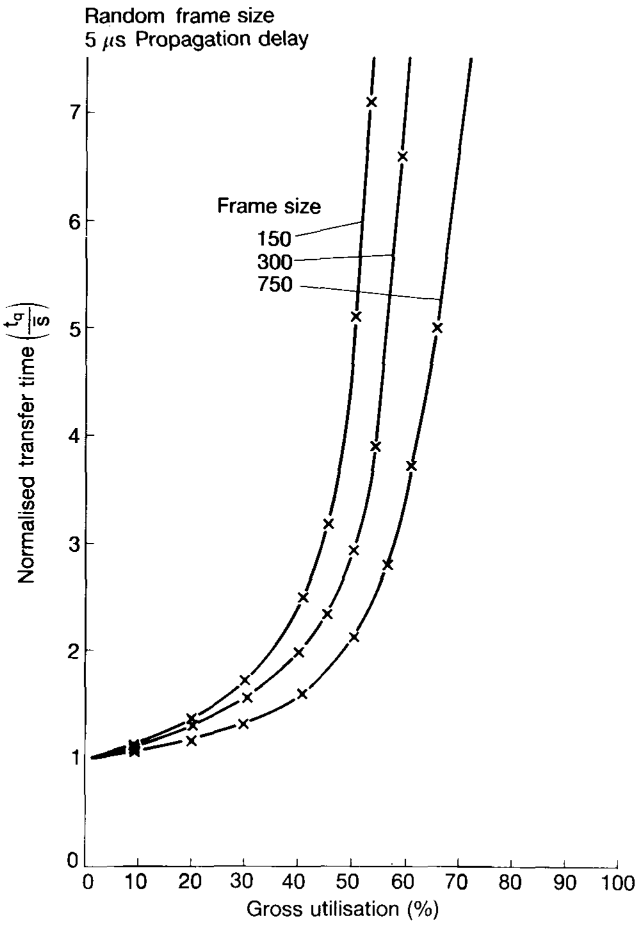


Fig. 3

A significant characteristic of a queuing system, in addition to the mean delay experienced, is the variability of that delay, and in particular the proportion of frames having abnormally long delays. Fig. 6 shows examples of the 95th percentile curve, compared to the mean delay. The position of the 95th percentile is close to that expected for a simple single server queue with the same service time distributions.

6.2 Asymmetrical transfer patterns

Some simulations were carried out with asymmetrical patterns of data flow, to see what effect this would have on Transfer Time.

– One station transmitting small frames, while the others are transmitting large ones. The small frames suffer more delay than they would on a

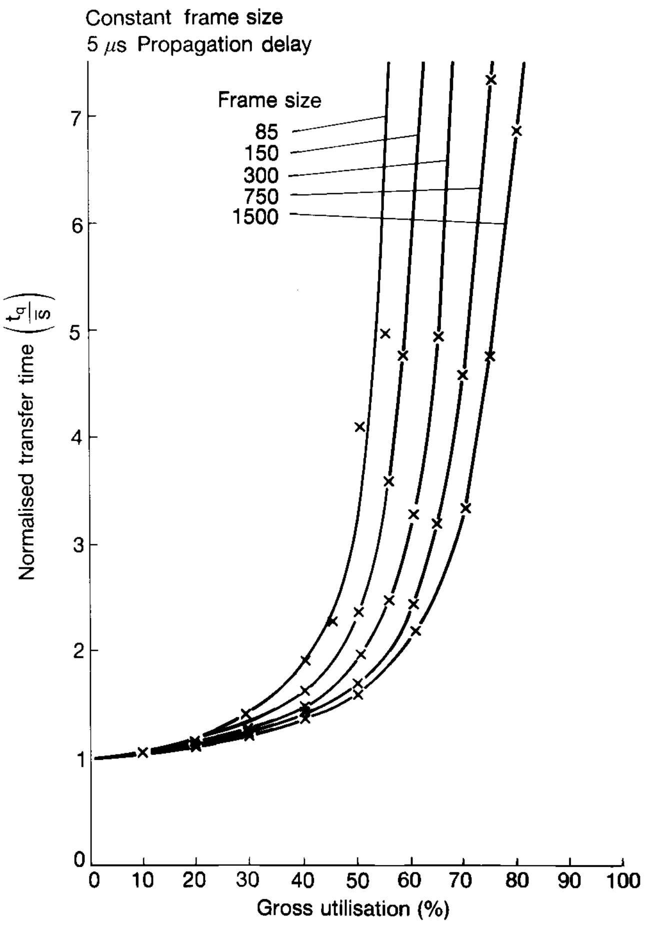


Fig. 4

similarly loaded network with only small frames, but the delay on small frames is still less than on large ones.

- One station transmitting large frames, while the others are transmitting small ones. Apart from the longer actual transmission time of large frames, both sizes experience very similar delays due to queuing and collision effects.
- One station transmitting half the total load, the other half being spread among many stations. No significant difference in Transfer Time is seen between the two types of station, or between this system and a symmetrical system under the same total Utilisation. The heavily loaded station did however show less variability in the delay experienced, and suffer relatively fewer collisions. This result is not what one might expect and this particular case would merit further study.

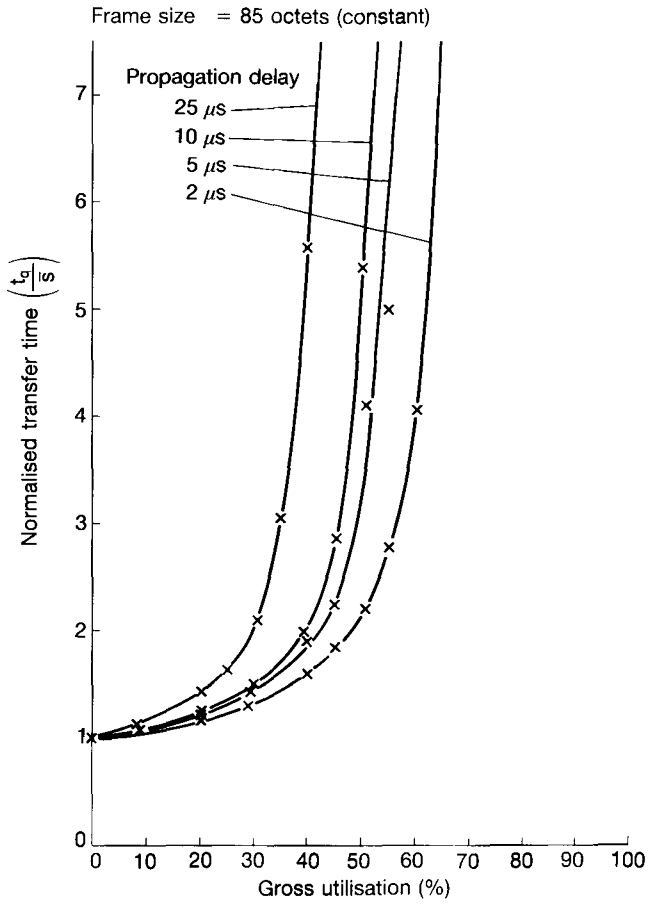


Fig. 5

These results indicate that the symmetry of load is not a significant factor in determining the Transfer Time of messages on the network; the total Utilisation and Mean Frame Size may be used in any estimate, without considering the distribution between stations.

6.3 Limited concurrency

Simulations were also conducted for systems with smaller concurrency, to determine the effect of this on Transfer Time.

– A system having a large number of stations, but only a single data message outstanding at once at each station: i.e. queuing effects within the transmitting station are eliminated. This system shows perhaps surprisingly little difference from the large concurrency model used for the

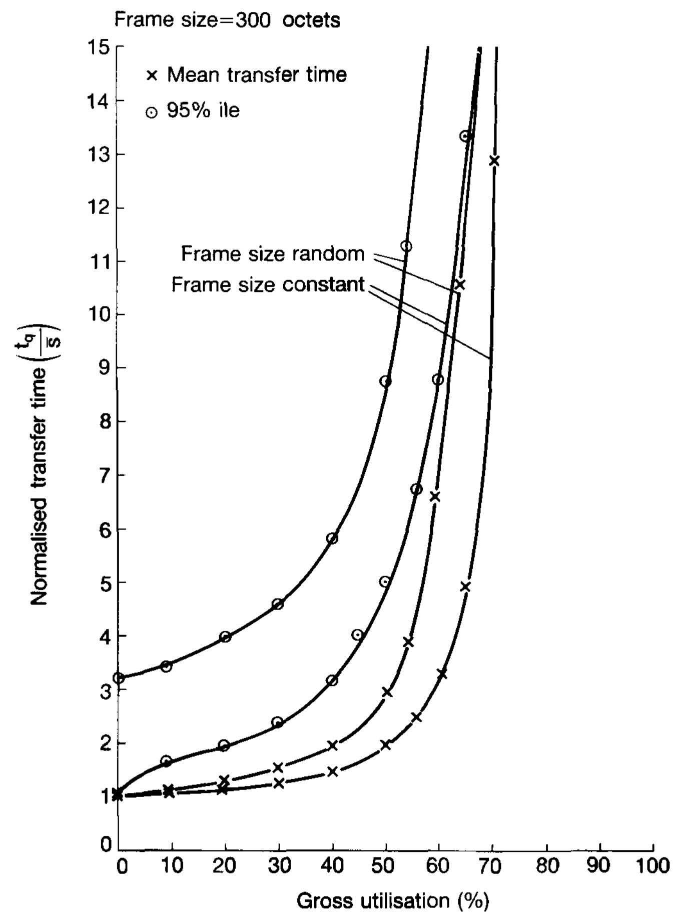


Fig. 6

remainder of the results. For large frames there is no significant difference observed, and for minimum size frames the Utilisation in the single stream case can be increased by up to 5% for the same overall delay.

- A system having only 5 stations, but with random (Poisson) arrivals. This is slightly worse than the 100 station case, reaching a given delay for about 3% lower Utilisation.

Though the performance of the system is affected by changes in concurrency, these results show that the change is small, and for practical purposes the results for a large number of stations, and high concurrency may be used.

7 Practical throughput limits

From the results plotted above it can be seen that all the curves are rising very steeply once a Transfer Time of 5 milliseconds, or a normalised time of

5, is reached. The practical consequences of this are:

- below a normalised Transfer Time of (say) 5 the OSLAN delay is negligible compared to other delays in the system. (Even for a 1500 byte frame 5 times the basic transmission time is still only 6 milleseconds.)
- above a normalised Transfer Time of 5 the curve is rising so steeply that it will very quickly outweigh all other sources of delay and the system will break down.

In summary the practical effect of OSLAN loading, taken in conjunction with other system delays, is that up to a certain point the OSLAN can be ignored, but that above that point the system runs into a solid limit. The practical sizing question, therefore, is to establish the position of that limiting value of Utilisation.

Figs. 7 and 8 plot this limiting value of Utilisation, taken as occurring at a Normalised Transfer Time of 5, against frame size and propagation delay. Curves are shown for random and constant sized frames. These diagrams show an area to the bottom right which is safe, and an area in the top left which leads to breakdown, with a set of curves defining the boundary under different conditions.

Fig. 9 combines the various boundary curves to define three areas of working: safe in all cases, definite breakdown, and a danger area in which the system may or may not work depending on the detailed conditions.

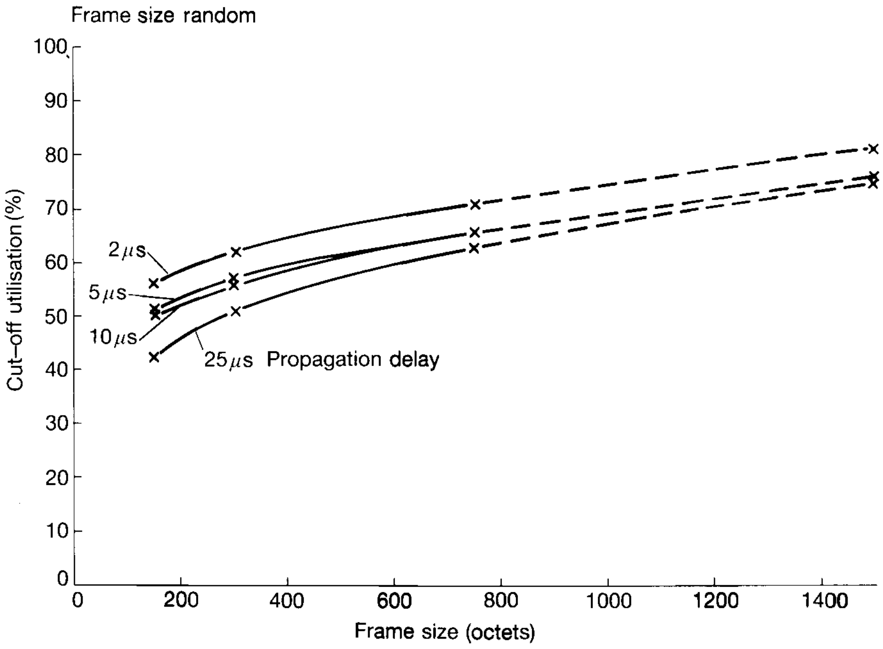


Fig. 7

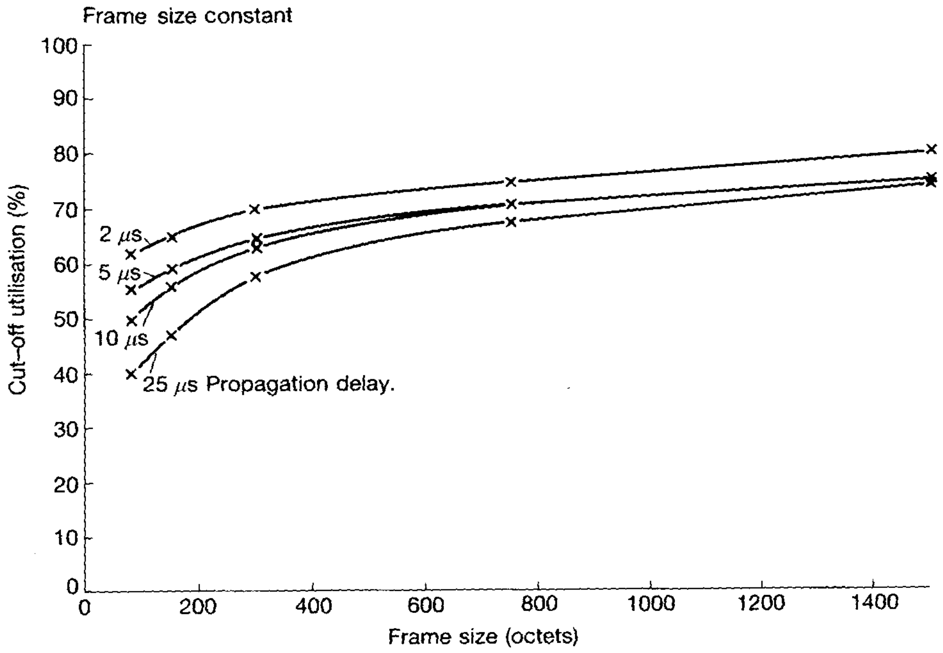

Frame size constant
Cut-off utilisation (%)
Frame size (octets)
2 μs
5 μs
10 μs
25 μs Propagation delay.
0
10
20
30
40
50
60
70
80
90
100
0
200
400
600
800
1000
1200
1400

Fig. 8

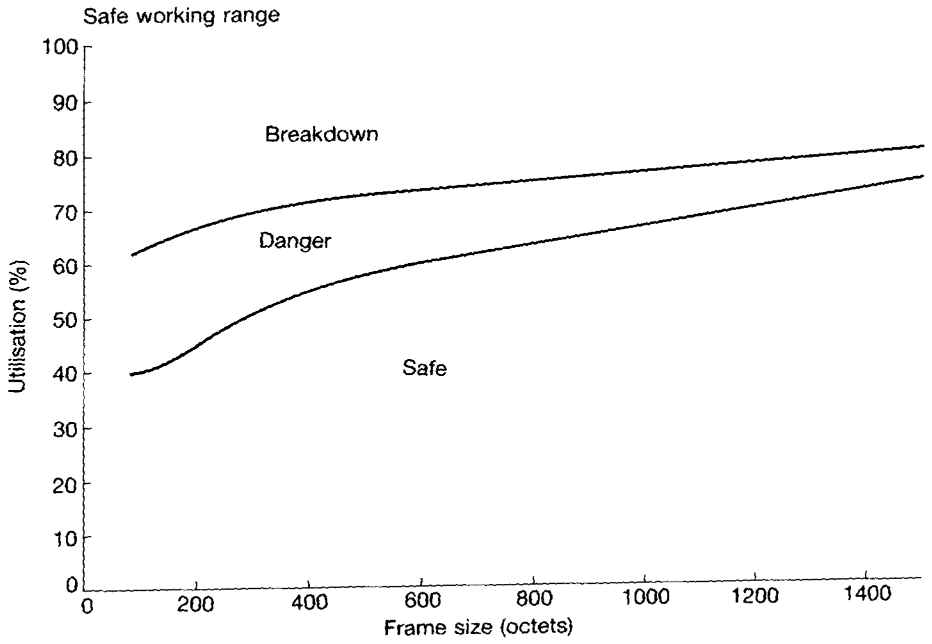

Safe working range
Utilisation (%)
Frame size (octets)
Breakdown
Danger
Safe
0
10
20
30
40
50
60
70
80
90
100
0
200
400
600
800
1000
1200
1400

Fig. 9

8 Interpretation of results

This section discusses various aspects of the main result represented by Fig. 9 and relates it to some practical cases.

8.1 *What happens under overload*

Under overload conditions the OSLAN reaches a saturation throughput and then continues to transfer this amount of data regardless of the actual load presented. This characteristic, of constant rather than reducing throughput under overload, is shown by the ESIM simulation, and by published experimental data[7]. The effect of such overload on real systems has not been investigated during these studies but the following can be expected:

- a proportion of frames requires higher level recovery, due to the backoff limit being reached. In ICL systems the ISO Transport Layer is responsible for this function. Simulation indicates that even under overload this proportion is only of the order of a few percent.
- much more significantly the excess traffic builds up in transmission queues, which theoretically would grow indefinitely. The effect of this will depend on the self-limiting actions of higher level protocols, and the end system's reaction to store overload.

8.2 *The limits in practice*

Looking at Fig. 9 it seems possible to conceive of cases in which the OSLAN will only support 40% Utilisation, which is a relatively low value. To put this in context it must be remembered that this is for the worst case of a network of maximum physical size, transmitting only minimum sized packets, and that this still represents a transmission rate of nearly 6000 frames/second. More typical Utilisation limits are in the region of 50% to 70%.

The practical capacity of OSLAN to support computer systems is illustrated by the following examples.

A file transfer system. If the OSLAN is used purely for bulk file transfer, performed in 1024 byte frames, with the Transport Layer protocol adding one Acknowledgement frame for every two data frames, the throughput achievable is as follows.

Gross size of data frame	$1024 + 47 = 1071$ octets
Gross size of acknowledgement	$= 84$ octets
Mean frame size	$((2 \times 1071) + 84)/3$
	$= 742$ octets
From Fig. 9 the safe Utilisation	$= 62\%$
which allowing for overheads	$= 745$ Kbytes/sec. of User data

An interactive system. If the OSLAN is used purely for interactive traffic with a message profile of 200 bytes inward and 1000 bytes outward, and with the Transport Layer acknowledging each frame separately, then the throughput achievable is as follows.

Gross size of inward Data frame	200 + 47 = 247 octets
Gross size of outward Data frame	1000 + 47 = 1047 octets
Gross size of Acknowledgement	= 84 octets
Mean frame size	(247 + 1047 + 84 + 84)/4 = 365 octets
From Fig. 9 the safe Utilisation	= 53%
which allowing for overheads	= 450 Message Pairs/ second.

In either of these types of system the performance limits of OSLAN are comfortably above those of the computer systems which rely on it.

8.3 A mathematical representation

A valuable technique to give a more precise representation of the OSLAN behaviour than that in Fig. 9 is to fit a purely empirical formula to the simulated data. This formula may then be used to predict the system behaviour, rather than going to the expense of running the simulator in each case.

Since the network is basically a queuing system, and since the curves obtained appear to follow the general shape of curves from queuing theory models, it is postulated that the OSLAN performance can be approximated by the formula for an *M/G/1 queue, with suitable parameters.

The formula for total time in the system, waiting and being served, for an M/G/1 queue[8]:

$$\overline{t_q} = \bar{s} + \frac{\lambda \overline{s^2}}{2(1-\rho)} \tag{1}$$

where λ is the mean arrival rate, ρ is the utilisation, and $\bar{s}$ is the mean service time. Normalising with respect to $\bar{s}$, this may be expressed as:

$$t_{norm} = 1 + \frac{\lambda\bar{s}}{1-\lambda\bar{s}} \cdot K \qquad \left[K = \tfrac{1}{2}\left(1 + \left(\frac{\sigma_s}{\bar{s}}\right)^2\right)\right] \tag{2}$$

where K is a pure constant dependent only on the form of the service time

*This notation defines a single-server queue in which the distribution of arrival times is exponential (M = Markovian) and that of service times is not specified (G = General).

distribution. For the two special cases of exponential (random) and constant service time, K equals 1 and 0·5 respectively.

It is further postulated that to obtain an approximation for t_{norm} as observed in the simulation, the actual value of $\bar{s}$ is replaced by the following function, λ and K remaining unchanged:

$$S_e = aL + bD + c \tag{3}$$

where L is the gross frame length, D is the propagation delay, and a, b, c are unknown parameters. A curve fitting algorithm applied to these formulae, using the values obtained during the simulations, gives the following values for a, b, c. (L is expressed in octets, and D in microseconds).

$$S_e = (0{\cdot}929L + 2{\cdot}47D + 21{\cdot}5)/10^6 \tag{4}$$

Combining this with equation (2) above allows the Normalised Transfer Time to be calculated for a given mean frame size, propagation delay, arrival rate, and size distribution. This formula is found to fit the set of points for which the Normalised Transfer Time equals 5, with a maximum error in calculated Utilisation of 6%. As an illustration of this Fig. 10 shows the curve for the Normalised Transfer Time equal to 5, directly from the simulation and from the formula calculated above. A curve from the theoretical

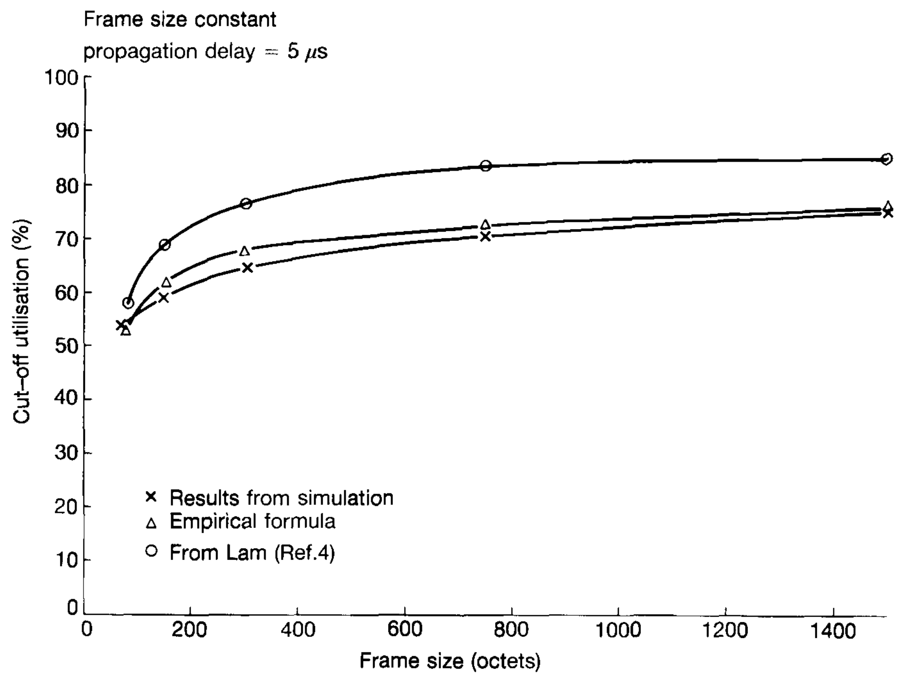


Fig. 10

treatment[4,5] is also shown, from which it can be seen that the approximations necessary to achieve an analytical solution do not allow very accurate results.

9 Conclusions

For a given combination of frame size, cable length, and other factors an OSLAN shows negligible transfer delay, compared to other sources of delay in the system, up to a certain value of Utilisation, but above this value the delay increases rapidly and the capacity hits a limit. Above this limit frames become subject to Transport Layer recovery and queues will build up, the exact system behaviour depending on higher level recovery and traffic limitation mechanisms.

The values of this limiting Utilisation are plotted in Fig. 9, which shows a safe zone, a breakdown zone, and a danger zone in which the exact parameters of cable length, frame size distribution, and concurrency will determine whether the system will work or not. An empirical formula is given to help establish the likely result in such cases.

If the limiting values of Utilisation are translated into throughput (in Kilobytes/sec or Message pairs/sec) for realistic traffic profiles, it can be seen that an OSLAN is capable of carrying a very high traffic load compared to the capabilities of the computer systems now being designed to use it.

Validation has shown that reasonable confidence can be placed in the results given, but further checking against experimental data is always desirable.

Acknowledgements

I would like to thank Peter Sawyer of ICL Bracknell, who originally wrote much of the simulation program used. Without the good start he provided I would not have been able to complete this work.

References

1 International Standard 8802/3.: 'CSMA/CD Access Method and Physical Layer Specifications'.
2 IEEE Standard 802.3.: 'CSMA/CD Access Method and Physical Layer Specifications'.
3 METCALF, R.M. and BOGGS, D.R.: 'Ethernet: Distributed Packet Switching for Local Computer Networks', Comm. ACM, vol. 19(7), 395–404 (1976).
4 LAM, S.S.: 'A Carrier Sense Multiple Access protocol for local networks', Computer Networks, vol. 4, 21–32 (1980).
5 MAGLARIS, B. and LISSACK, T.: 'An Integrated Broadband Local Area Network Architecture', Proc. 5th Conf. on Local Computer Networks, Minneapolis, (1980).
6 BUX, W.: 'Local-Area Subnetworks: A Performance Comparison', IEEE Trans. on Comm., com-29(10), 1465-73 (1981).
7 SHOCH, J.F. and HUPP, J.A.: 'Measured Performance of an Ethernet Local Network', Comm. ACM, vol. 23(12), 711–721 (1980).
8 KLEINROCK, L.: 'Queuing Systems', 2 vols, J. Wiley, (1975/6).

Experience with programming parallel signal-processing algorithms in Fortran 8X

A Wilson
ICL Services, Putney, London

Abstract

A brief summary is given of the array features that are currently planned for inclusion in Fortran 8X, the use of which will lead to shorter and more readable source programs that will compile into more efficient object code, particularly on vector or array target machines. These features were greatly influenced by the corresponding provisions of the DAP Fortran language, designed for and extensively used on the ICL DAP machines. Their use in signal processing is exemplified.

1 Introduction

In 1983 ICL began a project with RSRE Malvern that included the implementation of a multimode airborne radar system on the ICL DAP 2. In the course of this project it proved useful to represent certain radar algorithms in terms of the array features planned for inclusion in Fortran 8X.

From the programmer's point of view DAP 2 is virtually identical to DAP 1[2,3].

We now describe features of Fortran 8X that we found to be particularly useful in expressing the radar algorithms.

2 Summary of features

In this section we summarize the principal array features used in modelling the radar application[1].

2.1 Whole array expressions and assignments

The most crucial extension is that whole array expressions and assignments are permitted. For example the statement

$$A = B + C*SIN(D) \qquad (1)$$

where A, B, C and D are arrays of exactly the same shape is permitted. It is interpreted *elementwise*, that is the sine function is taken on each component of D, each result is multiplied by the corresponding component of C, added to the corresponding component of B and finally assigned to the corresponding position in A. Functions, including user-written functions, may be array-valued. All arrays in an expression or across an assignment must 'conform', that is have exactly the same 'rank' (number of dimensions) and 'shape' (set of lengths in each dimension), but scalars may be included freely and these are interpreted as being broadcast to a conforming array. Expressions are evaluated as a whole before assignment takes place.

2.2 *Array sections*

Wherever whole arrays may appear in expressions it is also possible to use rectangular subarrays called 'sections'. For example

```
A(:,1:N,2,3:1,-1)
```

consists of a subarray containing the whole of the first dimension, positions 1 to N of the second dimension, position 2 of the third dimension and positions 1 to 3 in reverse order for the fourth dimension. This is a bizarre artificial example chosen to illustrate the different forms. Of course, the most common use will be to pick out a row or column of an array. Thus the Ith row of A is A(I,:) and the Jth column is A(:,J).

2.3 *WHERE statements*

The WHERE statement provides a means of applying a conforming logical array as a mask to an assignment. For example

```
WHERE(TEMP.GT.100)PRESSURE = 2 * PRESSURE
```

assigns new values to those elements of the PRESSURE array which correspond to values of the TEMPerature array which are greater than 100. The effect is to 'double the pressure wherever the temperature exceeds 100'. All other values of the PRESSURE are unaffected.

2.4 *Array constructors*

A one-dimensional array of constants may be constructed using an 'array constructor' exemplified by [1, 2, 3] which is an array of length 3. Similarly [1:N] signifies [1, 2, ..., N]. Only rank one arrays may be constructed in this way, but higher dimensional arrays may be made from them by using the intrinsic function RESHAPE.

2.5 *Intrinsic functions*

All the Fortran 77 intrinsic functions (e.g. SIN, COS, EXP, ...) are extended in Fortran 8X to apply elementwise to arrays of any rank. In addition

Fortran 8X provides a wide range of new intrinsic functions which operate on whole arrays to perform common operations such as summation of the elements of an array (SUM), shifting the elements of any array cyclicly along a given dimension (CSHIFT), performing matrix multiplication (MATMUL), calculating a vector dotproduct (DOTPRODUCT), counting the true elements of a logical mask (COUNT), reshaping a one dimensional array into a multidimensional array (RESHAPE).

For our present purposes, we will indicate the workings of the CSHIFT and COUNT intrinsic functions.

2.5.1 The CSHIFT intrinsic function The intrinsic function CSHIFT(ARRAY,DIM,SHIFT) returns a cyclicly shifted version of its first argument ARRAY. The shift is carried out along dimension DIM of ARRAY by the amount SHIFT. The direction of shift depends on the sign of SHIFT. A vector has only DIM = 1. The column direction of a matrix is DIM = 1 and the row direction is DIM = 2. Elements shifted out at one end (of a vector or a given dimension) are shifted back in at the other end.

Some examples: CSHIFT([1,2,3],DIM = 1,SHIFT = 1) is [2,3,1].
CSHIFT([1,2,3],DIM = 1,SHIFT = −1) is [3,1,2].

The result of CSHIFT(X,DIM = 1,SHIFT = −1)

where X is the array

$$\begin{matrix} A & B & C & D \\ E & F & G & H \\ I & J & K & L \\ M & N & O & P \end{matrix}$$

is the array

$$\begin{matrix} M & N & O & P. \\ A & B & C & D \\ E & F & G & H \\ I & J & K & L \end{matrix}$$

2.5.2 The COUNT intrinsic function The intrinsic function COUNT(ARRAY) returns the count of the number of true values of the logical array ARRAY. Thus COUNT(V .GT. 0) is the number of positive numbers present in the vector V. Another example: using the fact that the elements of the vector X(1:N−1)*X(2:N) are precisely the products X(1)*X(2), X(2)*X(3), ..., X(N−1)*X(N) it follows easily that the number of sign changes in the sequence of numbers X(1),X(2), ... ,X(N) is given by the value of

```
COUNT( X(1:N−1)*X(2:N) .LT. 0 ).
```

3 Some calculations relating to a radar algorithm

3.1 Introduction

On DAP 2 it is most natural to process about 1000 bursts of radar signals in parallel.

Leaving aside input and output of the bursts, the RSRE/ICL radar application algorithm performs that process in three stages: spectrum analysis, CFAR detection and target identification. Each stage was initially specified by RSRE in documents employing mathematical symbols, radar terminology and plain English. Later, each stage was 'echoed' by ICL in documents summarizing the implementors' understanding of the algorithm in terms of Fortran 8X statements.

Here we present some of those Fortran 8X statements together with brief descriptions of the parts of the radar algorithm which they represent.

3.2 The radar algorithm

3.2.1 Spectrum analysis The spectrum analysis phase of the algorithm performs the FFT algorithm in parallel on the 1000 or so bursts of radar data. Each burst is a vector of 64 complex numbers. The bursts are held one per column in a total of five complex arrays each of 64 rows and up to 200 columns.

For purposes of exposition we will single out one of these arrays of complex numbers (giving it the name and dimensions SIGNALS(64,200)) and consider the problem of programming the FFT algorithm *in parallel* on (the 200 bursts represented by) the 200 columns of SIGNALS. More precisely, we ask how we can represent this parallel process in terms of the Fortran 8X array features described above?

The kernel to the solution of this problem in 'parallel programming' is the observation that, whereas the familiar notation SIGNALS(I,1) allows only the manipulation of one scalar at a time as I varies (namely the 64 complex scalars which inhabit the first column of the array SIGNALS), the Fortran 8X notation SIGNALS(I,:) permits the manipulation of 200 scalars at a time as I varies (namely the entire Ith row of SIGNALS).

Therefore if we write an ordinary serial FFT algorithm to operate on the elements of the first column (denoted by SIGNALS(I,1)) we can obtain the desired parallel algorithm simply by replacing each of those elements by the whole row of 200 corresponding elements (denoted by SIGNALS(I,:)).

The elementwise definition of the value of Fortran 8X array expressions then guarantees that the program we have described produces in parallel the complete array of transforms of the 200 columns of SIGNALS.

In brief the Fortran 8X program used to represent the spectrum analysis of some 1000 bursts of radar signals in parallel consists of a *serial* FFT algorithm applied to the *rows* of an array.

3.2.2 Spectrum analysis (continued) Suppose now that the complex array SPECTRA(64,200) contains the result of the application to the array SIGNALS of the parallel FFT algorithm just described.

Thus SPECTRA is the array to be processed next, in the CFAR phase of the radar algorithm.

However, before proceeding with the CFAR algorithm, we require to make an elementwise calculation on SPECTRA which is designed to replace the complex frequencies it contains with a measure of their absolute magnitudes.

Specifically, we compute a matrix of non-negative real numbers called SPIKES(64,200) by means of the following Fortran 8X statements:

```
SPIKES = 0
WHERE(ABS(SPECTRA).GT.1)SPIKES = LOG(ABS(SPECTRA)).
```

The first statement sets all SPIKES to zero. The second records (as positive spikes) the logarithm of the modulus of each complex frequency whose modulus exceeds 1.

3.2.3 CFAR (Constant False Alarm Rate) The CFAR processing now proceeds on the SPIKES array, which may usefully be thought of as a radar signal frequency relief map.

3.2.4 Nearest neighbour averaging We first apply to SPIKES a simple two dimensional filter which replaces each individual 'spike' with the average value of its 4 nearest neighbours. The boundary values necessary to this process are supplied by cyclic periodicity.

It may come as a mild surprise that such a filter is achieved in Fortran 8X without the use of either a looping construct ('DO loop') or the embedding of the SPIKES array in a larger array to provide the boundary values.

In fact the CSHIFT function automatically supplies the necessary wrap-around boundaries. Thus, using a 64 by 200 array called FILTERED to receive the result, we compute:

```
FILTERED = 0·25*(CSHIFT(SPIKES,DIM = 1,SHIFT =  1)
               + CSHIFT(SPIKES,DIM = 1,SHIFT = -1)
               + CSHIFT(SPIKES,DIM = 2,SHIFT =  1)
               + CSHIFT(SPIKES,DIM = 2,SHIFT = -1))
```

Here the 4 copies of SPIKES are respectively first shifted cyclicly South, North, West and East before being added together and averaged.

3.2.5 Peak detection The next calculation shows how easy it is in Fortran 8X to produce a map of 'interesting' points by simply writing down the 8X expression that visibly defines the 'interesting' condition.

For example a mask that locates precisely those frequency magnitudes that tower over their nearest neighbours to the extent of exceeding twice the neighbours' average value is computed by the statement:

```
PEAKS = SPIKES .GT. 2*FILTERED.
```

Thus PEAKS is a logical mask which locates those SPIKES that stand out in high relief.

The concluding stage of our illustrative calculations will identify bursts of radar signals which are deemed to have been reflected from probable targets.

3.2.6 Target detection It is considered likely that a given burst of signals arose from a target only if the burst produced at least three peaks. Thus we need to count the number of peaks in each column of the array PEAKS.

The function COUNT can be simultaneously applied to all the individual columns of the array PEAKS by means of the expression COUNT (PEAKS,DIM = 1), which returns a rank 1 integer array containing 200 counts. The parameter DIM indicates that counting occurs along dimension 1, i.e. down each column.

Finally then a logical vector of length 200 which locates the likely targets may be computed simply by the statement:

```
TARGETS = COUNT(PEAKS,DIM = 1) .GT. 3.
```

Thus each true element of TARGETS indicates that the corresponding column of SPECTRA contains three or more frequency values thought likely to have been reflected from a genuine target.

3.2.7 Summary of statements A summary of the radar processing statements following the computation of the array SPECTRA is as follows:

```
SPIKES = 0
WHERE(ABS(SPECTRA).GT.1)SPIKES = LOG(ABS(SPECTRA))
FILTERED = 0·25*(CSHIFT(SPIKES,DIM = 1,SHIFT =  1)
               + CSHIFT(SPIKES,DIM = 1,SHIFT = -1)
               + CSHIFT(SPIKES,DIM = 2,SHIFT =  1)
               + CSHIFT(SPIKES,DIM = 2,SHIFT = -1))
PEAKS = SPIKES .GT. 2*FILTERED
TARGETS = COUNT(PEAKS,DIM = 1) .GT. 3.
```

The above 5-statement computation in Fortran 8X expands to about 30 statements of ordinary Fortran and is then considerably more difficult to understand.

Conclusions

Fortran 8X proved to be very useful as a means for ICL to 'echo' to RSRE its comprehension of the component parts of the developing radar algorithm in a highly condensed yet easily understood manner; the capacity of Fortran 8X to express operations on whole arrays also greatly facilitated a natural mapping of the radar algorithm onto the array-structured DAP.

References

1 SIMPSON, P., MERRIFIELD, B.: 'Real time applications of a distributed array processor', Proceedings of the Second International Conference on Vector and Parallel Processors in Computational Science, editors J. K. Reid and I. S. Duff, Elsevier, 1985.
2 GOSTICK, R.W.: 'Software and algorithms for the Distributed Array Processor' ICL Tech. J. 1979, 1(2), 116–135.
3 HOWLETT, J., PARKINSON, D., SYLWESTROWICZ, J.: 'DAP in Action' ICL Tech. J. 1983, 3(3), 330–344.

Book review

Memoirs of a Computer Pioneer

Maurice Wilkes FRS. MIT Press 1985

The spread of computer technology has been so fast and all-pervasive, influencing so radically the concepts we use and accept in so many fields of human thought and activity that the greatest interest attaches to how and why computing began as it did. Was it a consequence of scientific and engineering developments in radio, television and radar, or were these merely the technologies necessary to give effect to ideas formulated earlier by say Babbage, Turing, von Neumann, and others? To what extent were imaginations stimulated by the prospect of realising earlier abstract ideas in practical terms?

Wilkes' book may help to answer such questions. About half of it is devoted to his family background and education both at school and at Cambridge, where he was a scholar (and later a fellow) of St. John's College. To judge from the book, he seems to have been fascinated equally by mathematics and radio technology and to have acquired considerable expertise in both well before he left school, where he had his own call sign. He was lucky to have gone to Cambridge in the thirties, when ionosphere research was flourishing. The fact that it used radio techniques as well as mathematics gave him great scope in research at the Cavendish Laboratory under Mr J A Ratcliffe before war started.

During the war he worked on a variety of radar projects for both the Army and, later, the Air Force, and records the inspiration he got during this time from direction by some of the leading physicists of the day including Professor Cockcroft. The work for the Air Force was at the same establishment at Malvern as that of Freddie Williams, who later initiated computer research at Manchester. But the motivation was quite different; although Maurice undoubtedly enjoyed the practical engineering that went into EDSAC, his aim was to build and use a better computing engine, which would replace the differential analysers installed in the Cambridge Mathematical Laboratory of which he was Director. These had come from Manchester with Professor Hartree whose constant encouragement he records.

Before EDSAC started working, his team, including David Wheeler and the late Stanley Gill, had already decided how to set down programs with addresses in decimal notation to be fed into the machine from paper tape and had planned to combine independently written sections of a program, using Wheeler's ingenious sequence of "initial orders" wired onto a telephone Uniselector switch. This contrasted strongly with the "wet finger" method used at first in Manchester to load a program a bit at a time, which had allowed Professor Williams to demonstrate a computer executing a stored program some months earlier.

The chapter describing the construction of EDSAC I from 1946 to 1949 is most fascinating. The story is told in a very modest and straightforward way, while Wilkes' debt to others for ideas is fully acknowledged, especially to the course he attended at the Moore School of Electrical Engineering of the University of Pennsylvania in 1946. The strong tradition in Cambridge physics at that time of linking research to teaching led to courses in programming being set up very soon after EDSAC began to work. A regular series of colloquia was launched open to all comers. Everyone interested in computing in the UK at that time attended. They included not only university and government research workers, but also people from EMI, English Electric, Elliott Brothers, LEO Computers, British Tabulating Machine Co., and Powers Samas. At the same time a computing service for the whole University was initiated.

Early applications were by Kendrew for calculating the structure of myoglobin from X-ray data and by Ryle to map the radio sources of the cosmos. The power of EDSAC I was soon seen to be inadequate. For EDSAC II, a more powerful machine to supersede EDSAC I, Wilkes proposed and successfully introduced, microprogramming – a concept that has lost some of its original clarity, but nevertheless continues to have a powerful influence on computer architecture today.

The contributions Wilkes made to the British Computer Society (of which he was the first President) and to commercial development of computers in the UK generally are less prominent. His influence on the LEO project in its earlier formative phases is fully described however, starting with the visit by John Simmons and Raymond Thompson in 1947. It is pleasant to recall the continual help the LEO team received at all times, but especially at the beginning from him and from his tireless assistant Eric Mutch. He does not refer to the continued advice and encouragement he went on giving to anyone who asked as long as he was at Cambridge.

To sum up, the book is an absorbing account of the early history of computers, to which it makes a notable contribution, and a penetrating commentary on later development up to the present. Rightly, it gives most detail about the earliest years when, as one now sees, key decisions were made that shaped the future. Everyone working with computers will enjoy reading it.

J.M.M. Pinkerton

Notes on the authors

N.C. Austin

Noel Austin Joined English Electric Leo Computers Ltd. in 1964, which company became part of ICL in 1968. He has had extensive experience of the application of computer-based techniques to the solution of business problems in a variety of industries and has developed specialist skills in business strategy, market planning and information systems strategy. Since mid-1984 he has worked as a Management Consultant within ICL's Management Support Business Centre. Among other qualifications he has a BA degree from the Open University and is a Member of the British Institute of Management.

E. Babb

Ed Babb obtained qualifications in Electrical Engineering and Computer Science from Imperial College. He then researched adaptive pattern recognition systems at Cambridge University on secondment from Hawker Siddely Dynamics. From about 1971, working in ICL, he researched speech recognition and information retrieval. His work on CAFS covered storage structures, architectures and query languages. He is now studying the application of mathematical logic to business and is currently the manager of the logic language project in the Systems Strategy Centre at Bracknell.

R.J. Bunyan

Roy Bunyan joined ICL in 1966 and has spent most of his career in operating systems support and development. He has been working in DRS development since its inception and has been managing the DRS300 Advanced Workstation for the past two years.

C.R. Crawford

Chris Crawford is an O & M analyst who joined ICL in 1972, having previously been employed by the BBC and by Davy Ashmore Engineering. He has worked in Group Information Services and its predecessors in various systems and management roles, with the occasional secondment to other Divisions of the company. In late 1985 he transferred to ICL (UK) Ltd. as the Strategy and Control Manager of the Management Services function.

C. Eden

Colin Eden graduated from Leicester University with a degree in engineering. Following his first degree he spent some time working with ICL before returning to University at Southampton to do a PhD in operational research. After a further stretch in industry as an operational researcher and management consultant he became a lecturer in Decision Analysis at the University of Bath. He is currently a Reader in the School of Management and director of the Strategic Decision Support Research Unit. He is a consultant to several international organisations and is the author of three books and many papers written on the subject of 'problem solving in teams' and 'strategic management'. He has been at the forefront of the move to 'soften' operational research studies by developing a greater synergy between mathematical modelling and organisational psychology.

P.D. Hall

Peter Hall is a Senior Business Consultant in ICL's Management Support Business Centre. He has undergraduate and postgraduate degrees in Electrical and Control Engineering from Cambridge University and a Masters degree in Management Science from Stanford. He served in the Royal Navy where his assignments included managing the development of radar missile systems. He has both taught and done research in nuclear science and has worked, mainly in the USA, in the Polaris project. His management experience in ICL has included computer services, software development, marketing and business planning; he is now researching in top-level decision support systems in collaboration with the Decision Analysis Unit of the London School of Economics and with the Management into the Nineties programme at the Sloane School of Management, Massachusetts Institute of Technology. He is a network member of the Tavistock Institute of Human Relations.

R.W. Jones

Roy Jones is a security consultant in ICL's Defence Technology Centre. He has a degree in Modern Languages from Nottingham University and has worked as a systems designer since 1956. He has been concerned with secure systems and the use of encipherment in computer architecture since 1975, first as a consultant with CCTA and then, since 1977, within ICL. He is a member of the BSI committee IST 20, whose task is to produce standards for the use of encipherment, and of the corresponding international committee ISO/SC 20.

C.J. Martin

Christopher Martin has degrees in Computer Science and Management Studies. His research interests include behavioural aspects of the implementation of computer systems in organisations.

A. Maynard-Smith

After graduating from Cambridge University with a degree in physics, Tony Maynard-Smith joined English Electric Computers, shortly to become part of ICL. He started work in a group designing communications controllers and terminals and has continued in this area for most of his career. He spent a considerable period in the Specials group in ICL working on a wide variety of networking systems. More recently he has been closely involved with the company's OSLAN developments and is now a consultant in STC Network Systems.

Professor M.S. Scott Morton

Michael Scott Morton is Professor of Management Science, Sloan School of Management, Massachusetts Institute of Technology. He is a specialist on the problems of developing computer systems to support managerial decisions and is active also in the field of co-ordinating corporate planning and control systems with management information systems. He was an IBM Fellow at Harvard University, where he received his Doctorate. His industrial experience includes membership on the boards of directors of several corporations and active consulting relationships with major corporations, including ICL. He is author of the book Management Decision Systems and co-author with John F. Rockart of Computers and the Learning Process; and with Peter G.W. Keen of Decision support Systems: an Organsational perspective.

S.J. Pollard

Stuart Pollard joined ICL and the Information Systems discipline in 1970, leaving in October 1986 for James Martin Associates to pursue a career in IS consultancy. For the majority of this time he has acted in a variety of application development roles – from programmer to systems manager – within Group Information Services and its predecessors. In recent years he has concentrated on the formulation of IS strategy and the exploiting of new products and methods.

Professor P.M. Stocker

Peter Stocker, a physicist by education and first experience, was Departmental Manager for scientific and technical applications with Ferranti Computers (later part of ICT) from 1961 to 1966. He then was appointed Director of the Computing Centre at the University of East Anglia, Norwich, where he now is also Professor of Computing Science. He has a special interest in databases and related areas, in which he acts as a consultant.

P.W. Veasey

Philip Veasey received an M.Sc. in Mathematics from the University of Warwick in 1968. As well as systems development with a Coventry engineering company, his experience has spanned seven years selling computer systems with Burroughs and Microdata and five years teaching in Third-World universities. Before joining ICL in February 1985, he was responsible for marketing Project Management software in South-East Asia. This experience led to the strong planning orientation which has influenced his work since joining the Strategy and Technology unit of GIS.

A. Wilson

Alan Wilson received a Ph.D. in mathematics in 1958 from Rice University and joined the faculty of the University of Pennsylvania in 1959. In 1961 he was a National Science Foundation Research Fellow at Imperial College London and joined Ferranti Computer Division in 1962. He was responsible for inhouse Computer Science training of Ferranti graduate intake and customer programming courses for the Atlas 1 and 2 computers. He has led three Fortran compiler development projects and managed the ICL Software Assurance activity. Since 1977 he has represented ICL to the American National Standards Institute (ANSI) Fortran Standards Committee. He is now a Specialist Consultant in ICL Services, working in the area of parallel processing and automated reasoning.