Information Resource Utilisation: Accessibility Based On Concerns

Richard Rolfe

Keynes College

Submitted For The Degree of Doctor of Philosophy
University of Kent at Canterbury
September 1991

Information Resource Utilisation: Accessibility Based On Concerns

Richard Rolfe
University of Kent at Canterbury

Abstract

The utilisation of the information resource is a key requirement of an effective information system within the organisation. With computing technology generating increasingly complex applications but with an increasing spread of computing contactability through the organisation, an inherent conflict is presented which can only worsen as hypermedia and multimedia applications are brought to the organisation. This work identified user participation as the key ingredient in improving information resource utilisation. A two pronged approach was taken: initially a field based investigation of the utilisation of a complex information system within a large organisation and secondly the development of a user interface environment to tackle the issues raised.

The fieldwork provided evidence to suggest the level of accessibility to Health Service information left wide scope for development. Several variables were identified as having a key relationship with information resource utilisation. Of these complexity predominated, both in terms of the database structures in the application and the functionality provided through the interface itself.

The prototype focused on this complexity issue directly, providing a user driven interpretative environment in which both user and organisation based information can be combined to form a dynamic, user based semantic model. Such a graphical representation evolves in paralell with user requirements, enabling the user to manipulate his/her own view of the application for the expression of database queries. The basis for this interpretation is the generalised conceptual model which is captured from the data modeller during the initial application data modelling process. This view forms the basis for user accessibility, acting as an initial interpretation which the user tailors towards a personal interpretation using 'concerns'. The provision of such an environment promotes user accessibility and hence utilisation, being personalised by the individual to the task at hand.

Thesis Overview

The role of computing within the organisation has changed dramatically in the past decade, moving away from the backroom technical support function towards front line management decision support. As computer technology has filtered up through the organisation towards senior management decision support, information systems that were previously perceived as a necessary but undesirable bureaucratic overhead have now been reassessed in terms of a new economic asset on a par with more traditional assets such as stock, debtors and goodwill. In recent years organisations have invested considerable resources in pursuit of perhaps an unquantifiable benefit - improved decision making through utilisation of this information resource.

The convergence of increasing processing power, mass storage technology and accessibility to diversified sources of information through integrated networks has facilitated generation of increasingly large and detailed organisationwide applications, for example the capture of increasing semantic information for database construction. A related effect due to the falling cost of computing technology, has been a diversification in the range of staff coming into contact with computer based applications, now encompassing non computer specialist personnel.

This conflict is growing rapidly. Swift advances have been made in the development of hypermedia and multimedia systems, building progressively larger more complete sources of information which include high resolution pictures, graphics and sound. Ultimately these approaches will encounter problems of falling utilisation as an increasing wider non computer specialist user population wrestles with these increasingly complex applications.

This thesis has focused through an applied perspective on this widening gap between non computer literate specialist personnel and the increasingly complex computer based applications which they are required to manipulate. This work has taken a two pronged approach to this issue: initially a field based investigation into the utilisation of a complex information system was carried out and secondly building on the gained experience, a user interface environment was developed to tackle this issue directly.

The fieldwork investigated utilisation of a conceptually complex information system set in a large organisation. The Health Service has undergone significant change, particularly in terms of information provision to decision makers, moving from paper based to computerised patient administration systems(PAS). With the continuing investment in development and implementation of such technology based information systems, the question of system utilisation takes on an increasing practical significance. This is especially true of the Health Service whose competing demands for patient care and administration detail under cost containment pressure can only increase in the coming years.

Trice & Treacy have identified utilisation as an intervening variable between user performance and the information technology. This work has identified a second intervening variable - that of accessibility to the information resource.

Supporting evidence is provided, suggesting the level of accessibility to Health Service information left wide scope for development. In particular the issue of user participation in determining the level of accessibility is identified as a combination of

- user awareness and practical interaction issues
- complexity of interface provision
- complexity of data model structure

It is only in recent years that the important role of the user interface and provision of human computer interaction have been recognised as fundamental ingredients in successful application development. Previously such concerns were an additional burden for the systems programmer and were consequently downgraded to an afterthought, once the 'core' processing routines had been established. The exponential increase in contactability with computing through the organisation has forced a reassessment of the role of the user and subsequently interface design in software development, for example increasing use is being made of rapid screen prototyping tools for system specification.

The success of the graphical user interface recognises this shift in emphasis - the user should not conform to the machine - rather the machine should conform to the user. However the role of the user interface can only go so far in tackling the issue of user accessibility. With any form of HCI, the user is still required to labour at the interpretation, that is the gulf of evaluation. This is especially true of database applications where to gain maximum benefit, the user is required to develop ad hoc requests for information. However in doing so the user is still required to manipulate the defined structure of the database directly.

From the fieldwork, such a requirement was identified as the key source of poor application utilisation. Such database structures are constructed by a systems modeller, that is someone other than the final end user. Consequently they are a logical, fixed past interpretation of the application domain. The user is therefore required to convert their interpretation of the information request into the formal interpretation required of the application and express this within a formal query syntax.

This work has tackled these interlocking aspects of application utilisation: database structural and interfacing complexity, by promoting user participation. The basis of this has been to involve the user, providing a graphical interpretative environment which can be manipulated to form a personal interpretation of the semantic structure of the database application. A key vehicle in achieving this is the concept of a concern: a graphically confined environment in which the user builds an interpretation of the semantic structure of the database application based on personal interpretation relating to the issue at hand. The basis for this graphical environment is the generalised conceptual model from the data modelling process.

Introduced is the prototype Query By Concern (QBC) which illustrates this approach. Both organisational and individual interpretative information can be combined with the base conceptual model into user concerns. Comprised of two components, the prototype enables graphical definition of the generalised and specific conceptual models by the systems modeller from which additional semantic information is captured during the mapping process.

This graphical representation then forms the basis for a dynamic end user interpretation of the information resource, evolving in paralell with user requirements. The provision of such an environment promotes user accessibility and hence utilisation, being personalised by the individual to the task in hand.

The thesis is presented in seven chapters. Chapter One discusses the changing role of information in the organisation, identifying the importance of accessibility in achieving the effectiveness and efficiency of this organisation asset. Chapters Two and Three present the fieldwork conducted with the Health Service investigating utilisation of the information resource. Several key variables were identified: of these complexity predominated in terms both of the database structure implicit in the application and the functionality provided through the interface itself. These issues are discussed in a theoretical context in Chapter Four and a prototype environment to address these issues is introduced and discussed in detail in chapters Five and Six. The final chapter, chapter Seven, provides a summary of the work achieved, possible enhancements and the future directions in which the work can be carried forward.

Acknowledgements

This research was undertaken during the period 1987-1990 as part of the Economic and Social Research Council Doctoral Programme in Applied Computing (Social Sciences). An initial year (1987/88) was spent in the taught component of this programme.

Thanks and warm gratitude must be extended to all those who made this work possible. Firstly thanks must go to those who have supervised me: Elizabeth Oxborrow, Mike Murphy, Steve Vickers, with Zarine Kemp and Brian Rutherford seeing it through to the end.

This work would not have been possible without the Computing Lab, Lynx and Mark Russell plus Unix support Crew with their semantic links, and Christine Wilson with WordPerfect. Thanks must go to all those in the Health Service who enthusiastically participated in this work.

Outside of the University, thanks must go to Nr 4 and its various inhabitants: George Pankievitch, Pete Thurston, Danuta Orlowska, Bruce Ingram, Penny Lawrence, Tim Goodwin, The Patons, Joanna and Jammy. Thanks also to UKC Trogs, especially McCamplin, McClusky, George, McHalliday, Martin, Niaomi, Sinead, Steve, and many others (especially Louise) with whom I have become muddy, wet and cold on numerous occasions.

Thanks to Dad and Beryl for letting me do my own thing and to Viv for all the aerobics and shiatsu which has kept my body and soul together. Lastly special thanks must go to Dr Elizabeth Oxborrow who started me off on this trail and has been a constant source of support and guidance, keeping me just about on the rails.

Presently a bell rang in the signal box and a man came running, "James is of the line - the breakdown train quickly", he shouted. So Thomas was coupled on, the workmen jumped into their coach, and off they went.

Thomas worked his hardest. "Hurry! Hurry! Hurry!" he puffed, and this time he wasn't pretending to be Gordon, he really meant it.

"Bother those trucks and their tricks" he thought, "I hope poor James isn't hurt.

They found James and the trucks at a bend in the line. The brake van and the last few trucks were on the rails, but the front ones were pilled in a heap; James was in a field with a cow looking at him, and his driver and fireman were feeling him all over to see if he was hurt.

...

They left the broken trucks and mended the line. Then with two cranes they put James back on the rails. He tried to move but couldn't, so Thomas helped him back to the shed.

The Fat Director was waiting anxiously for them.

"Well Thomas", he said kindly, "I've heard all about it, and I'm very pleased with you. You're a Really Useful Engine."

"James shall have proper brakes and a new coat of paint, and you -------- shall have a branch line all to yourself."

"Oh Sir!", said Thomas , happily.

<u>Thomas The Tank Engine</u>

The Rev W Awdry, Heinemann Press 1957

Table Of Contents

CHAPTER ONE

The Changing Role Of Information

Contents

1.0 The Changing Role Of Information

The role of computing in society is changing. This chapter presents the development of a telematic or information society[1], tracing the development of computing technology from the shop floor into routine office functions and now further into executive decision support. This chapter focuses on the development of information as a corporate resource, standing alongside more traditional assets such as goodwill or stock. In particular, effectiveness of this information resource within the organisation and the surrogate measure of utilisation as a success indicator are considered. Related work in the field is discussed and the importance of accessibility to the information resource identified.

1.1 Introduction: Organisational Impact and Computing Technology

The application of computing technology as a vehicle to transmit information can be traced from typed print using the first manual and electric typewriters, and for numeric information into desktop adding machines and subsequently hand held calculators. The technology progressed through two stages. Initially each form of information was treated separately as an extension of the manual operation, for example through the provision of wordprocessors for textual communications and spreadsheet/numerical applications for numeric information. Towards the end of the 1980's, with increasing mass storage capacity and communication accessibility through local and wide area networks(LAN/WAN), computing technology progressed to providing the integration of information through distributed databases and desktop publishing systems, providing accessibility to organisation wide information in a way never before possible.

At each stage in this development cycle computing technology has brought both explicit and implicit organisational and social consequences[2,3]. For example the office environment of the sixteenth and seventeenth centuries was based on quill technology, remaining a highly skilled male preserve similar to the accounting offices of the 1960's. With a change in the technology towards electronic typewriter or calculator/computer, the job was deskilled and restructured, predominantly employing female labour, for example into typing pools or data entry sections. The technology itself can be regarded as neutral[a]; it is the implementation process that is value loaded, reflecting the goals and aspirations of those commissioning the technology[4,5]. Computer technology is unique in this respect providing an environment that can explicitly monitor and control, for example conveyor belt or numeric controlled lathe technology can improve the pace of work output but remains unable to directly monitor the operative. Computer technology, for example in the form of accounting transaction/data entry functions can effectively monitor, pace and evaluate the worker explicitly[6].

[a] This is an 'optimistic' approach, see Land F, *The Impact Of Information Technology on The Work Place* which is set against the pessimistic approach of Braverman, *Labour & Monopoly Capital, the Degradation of Work in the Twentieth Century*, where technology is a surrogate for management domination.

The first effects of computerisation on the organisation occurred at the shop floor level, targeting the mechanisation of routine tasks, aiming at increasing productivity. Prior to World War II the shop floor was dominated by 'hard' technologies requiring a highly skilled labour force. During the war, 'soft' technologies were developed, particularly the development of computer machine control. These soft technologies were rapidly introduced after the war, through the provision of numeric controlled processing, for example computer numeric controlled(CNC) milling and lathing machinery. Such technology represented a significant investment, providing increased flexibility and management choice in the control and management of the production process. In particular design and programming for production could now be transferred away from the shop floor to an office based function promoting management control, effectively deskilling the shop floor labour force.

This level of accessibility provided to the technology was a conscious management decision over centralisation or decentralisation of control[7], as equal programming facilities were available on the shop floor as well as remotely in the design office[8]. Communications across the user-office divide were then constrained to a formalised procedure of control tape transfers. By the end of the 1980's, accessibility to the technology returned to the final end user as increasing emphasis was placed on job or small batch production, requiring frequent local control for retooling and reprogramming[9].

This turnaround in end user participation closely parallelled implementation of computer technology in the office environment. Here the aim of implementing the technology was the same - to mechanise routine activity justified on the same belief that savings on labour costs coupled with increased throughput would exceed capital and implementation costs[10]. These systems were again technically complex and physically large, representing substantial capital investment for the organisation. Similar questions of control and accessibility arose with final user departments, for example accounting, losing control of information processing to a newly created central Data Processing(DP) department in a shroud of technology and jargon. Requests for information were directed via this department (for example via systems designed, installed and located within the DP Department) instead of simply accessing previously locally held paper files. Communications across the user-DP department divide were constrained to batching details, for example control totals and error reports.

For the office environment the turnround in end user participation came through the late 1970's and early 1980's as computer technology developed into real time/interactive processing, opening the door for direct user-application communications without the direct intervention of the DP department. Such installations remained a significant capital outlay and were therefore centralised in the DP department, with end users sharing resources via networked direct access. The DP department still had a tight reign on service provision both in terms of hardware provision and support, and the generation and maintenance of application software.

The late 1980's saw a phenomenal growth in the demand for the now renamed information technology(IT) from all sectors of society, particularly from commerce and industry. Increasing public exposure to IT, for example through bank automatic teller machines(ATM) and point of sale(POS) systems, broadened public awareness of the technology. This was then popularised through films such as 'War Games' and in general through the popular press, for example the issue of computer hacking and crime. Internal to the organisation user awareness of the availability of IT paralleled this dramatic increase. This was compounded with management facing a comparatively dynamic external environment, but constrained to computer based information systems fixed by the DP department and structured towards routine, usually financial based information. Managers began to require more than simply computerised information systems that gave little more information than its paper driven equivalent, demanding instead complete interrogative analysis and reporting systems[11] which supported rather than replaced their own judgement[12].

Demand for application development and maintenance within the organisation increased exponentially, leading to significant backlogs and delays of application development and software maintenance with the DP department[13]. Managers under increasing external pressure (for example usually in response to crisis management) began to search for effective solutions to their unmet data processing demands. The falling capital cost of personal computing hardware to well within line management operating budget constraints, provided a direct solution that usually did not require DP Department participation or authorization. Turnkey solutions were available using this PC technology providing line management with localised access to computerised information support via 'friendly', integrated packages on a small scale, for example Lotus123[b] and DbaseIII[c].

These tools enabled line management to circumvent the DP department, obtaining solutions more suitable to their localised requirements. Such freedom however entailed no organisation wide direction for IT, generating a patchwork of incompatible systems, file formats and maintenance agreements. The achievement of short run problem solving was often made at the expense of long run effectiveness, for example problems of staff turnover, inadequate documentation and failure to make adequate backups or maintenance agreements. Questions of data validity and data integrity began to arise as users stored and manipulated private copies of data, each aiming at providing an individual DP department 'in the small'. Many DP departments ignored these developments, concentrating on main stream corporate wide applications. Those which did identify these problems combined both implicit and explicit measures to regain some measure of control over the user population.

[b] Lotus123 is a trademark for the Lotus Development Corporation.

[c] DbaseIII is a trademark of Ashton Tate Corporation.

Explicit measures took on the form of corporate strengthening of the DP department through organisation wide controls or standards often backed up by refusal to support hardware and software products not authorised by them. Implicit approaches took on the form of end user computing(EUC) and information support centres that actively provide support and guidance for end user departments and individuals[14,15]. An example of a hybrid approach was the 'Shell shop' operated by Shell PLC offering for sale heavily subsidised computer technology to internal departments and users. Users were able to choose external suppliers and products but these did not attract organisational subsidy and were subsequently more expensive. This effectively limited user choice to certain products supported internally, ensuring hardware maintenance and software support within an organisational framework.

With the continuing fall in hardware cost and increasing availability of processing power, the organisation has increasingly focused on improving productivity in the less routine tasks to be performed - those performed by decision makers or information workers[16]. Such Executive Information(EIS) or Decision Support Systems(DSS) for 'executive mind support'[17] began to appear in the late 1980's exploiting the full colour high resolution screen graphics technology coupled with increasingly large processing and storage devices. Targeted at management with non routine, ad hoc decision making, drawing on multiple sources of information, this technology aimed at improving productivity in decision making and the management of resources[18]. In this manner the base of staff coming into contact with computerised applications is diversifying rapidly as new application areas are exploited. This situation is intensifying as information technology filters up through the organisation hierarchy, progressing away from routine information provision (or MIS) towards ad hoc information processing or DSS/EIS for senior management.

Information technology has moved from handling compartmentalised data to handling knowledge work derived from the overall organisation. The key now is to achieve the benefits of EUC, that is localised, direct accessibility and manipulation, without losing the data consistency and integrity achieved through exclusive use of a DP Department[19,20].

1.2 Information as a Resource

In a matter of two decades the available computing technology has progressed from expensive and clumsy to cheap, reliable and physically accessible[21]. This has brought great change into the organisation, what was once seen as a necessary bureaucratic overhead has now been reassessed as a new type of asset - that of 'information'. Rigid systems of reporting and control have been overtaken by the emergence of an information economy[22].

Many traditional resources available to the organisation are tangible, they can be bought, sold and transferred as required. The accessibility and value of such assets passes with transfer, diminishes with use and can only be revitalised through replacement. Information on the other hand is intangible, is difficult to quantify and remains abstract. Information can be bought and sold as required but its value may remain even though its property may be transferred. It cannot therefore wear out, but may become lost, obsolete or crowded out by other data. The value of such an asset in economic terms is based on interpretation or capability of use in decision making, it is therefore contextual, subjective and unquantifiable in nature[23,24].

During the late 1970's it became increasingly evident that the manipulation and control of information could provide distinct economic advantage. For example the airline flight reservation information systems of that time provided effective barriers to competition entry, providing a rich source of market research information and provided a mechanism to tie customers to a single carrier[25]. These developments marked the beginning of a shift away from managing the computer technology itself, towards the explicit management of the information stored within the technology[26]. Such concentration on information as a resource to be managed separately in its own right has been made possible through the convergence of database and communication technologies, providing a single rich information repository.

The focus of management and control of this resource has been derived from an aggregation of three previously independent disciplines: database administration, librarianship and data processing[27]. Database management provides tools for database validity and integrity control expressed through storage management schemes. Record management/librarianship provide methodologies for indexing and information husbandry tools such as sorting and searching. Data processing management provides methodology for structured processing of large volumes of data for example through structured systems analysis and design methodologies(SSADM) and general management of the data resource function, for example Nolan's staged maturity cycle[28]. These developments have enabled a movement away from the notion of a single technology serving a single user via a single access path, towards a convergence of technology and data into a single information resource with multiple distributed access throughout the organisation.

The philosophy behind this approach is to maximise value from the technology through the use of information, rather than minimising cost. However the value and use of information is difficult to determine, partly because there is no economic market, often being acquired by the end user for personal use. From the users perspective, value is contingent upon human interpretation and both capability and application of use. The resource is based therefore on human capacity for adding value - information resource management(IRM) is therefore managing the human potential for adding value[29]. Further detailed definitions for IRM are common place. Lewis[30] provides a succinct definition:

> 'the integrated utilisation of three basic resources in the pursuit of higher productivity: people, information and systems (whether paper or computer based)'.

King and Kraemer[31] suggest that the management of this information resource is based on two fundamental beliefs: that information systems are amenable to systematic formal control and that information is an economic resource, therefore open to rational economic management. With such an approach the task of designing, implementing and monitoring such information systems is assumed to be not only possible but fully exercised by management. This model draws closely from the rational comprehensive model(RCM) in that information systems are specified, implemented and monitored in an economic, rational manner. In reality such a task becomes enormous as information systems evolve and replicate, cutting across departmental and organisational constructions. Information is also intangible, providing little audit trail, being mostly transmitted along informal networks with little regard for organisation structure. The second basis suggests that information is an economic resource open to rational management. Information is by its inherent nature positional and therefore open to widely differing interpretation and manipulation. The organisation as a unit is however often restricted in its ability to capitalise on this available knowledge for reasons of politics[32], communication or organisational dysfunction[33]. It remains questionable therefore whether 'rational' management and systematic control of such an unquantifiable asset is practically possible to the degree suggested by the definitional requirements of IRM.

The overall aim of information resource management is simply put by Best[28]:

> 'to stop knowledge being lost to information'.

It is therefore the information and the technology that need to be managed to the benefit of the user, not just the information or the technology itself in isolation. One is contingent upon the other. Information and computing technology therefore provides a new organisational asset whose evaluation and control, presents management with many new challenges.

1.3 Information as a Successful Organisation Resource

Like any other asset within the organisation, if information is to be regarded as worth maintaining it has to be evaluated both in terms of efficiency and effectiveness, both in production and delivery within the organisation[34]. For example the efficiency of a lathe can be measured in terms of cost input, for example power consumed, with productive output measured in terms of quantity and quality of finished goods produced. Efficiency can be established by monitoring scrap levels incurred in the production process. Applying the same rationality to information or knowledge provision is more problematic. The input side to the efficiency equation can be established: the number of staff hours and capital required can be objectively measured as these remain physical quantities. Defining and subsequently determining the output of such systems is more problematic to achieve. Example problems include What constitutes a unit of information? How is the usefulness or value of this unit to be determined? and how is the effectiveness of such an asset to be measured?

Several authors have tackled this question of which Swanson[35] provides an overview. Raymond[36] evaluates several possible surrogate operational measures as a means to evaluating the success of the information resource. One approach would be to measure the incremental change in decision making performance of either the individual or the organisation in consequence of an increased information provision. However it is difficult to specify acceptable measures of the differential performance or to control for other external factors, for example changes in environment or market. Measuring change in system usage provides for a more objective unit of measurement with the machine itself recording usage statistics, commands issued and functions performed in an unobtrusive manner. On its own, increased machine usage is however not a guarantee that the system is being successfully used, it may for example, be that increased usage statistics are a symptom of users struggling to come to terms with an inadequate user interface. Raymond follows Ives et al[37] and others[38,39], concluding for user satisfaction as a surrogate measure for system success, measuring in particular user attitude toward information services within the organisation. Such an approach however remains subjective and unquantifiable, giving problematic comparability across different users. For example all users are totally different with differing levels of organisation experience, computing skill[40] and personality traits[41].

Zmud[42] introduces a model of system utilisation based on individual differences as an indicator of system success. Two paths to information system success are presented: cognitive and attitudal. The cognitive path is based on user cognitive behaviour and associated effects of information systems design, for example how information systems are perceived, organised and acted upon. The attitudal path is built around attitudal variables of the information systems user, for example personality, demographic and situational variables. Throughout the model the implicit assumption behind each available route is that increased utilisation leads to information system success. Mykytyn[43] and Bailey and Pearson[44] approach this directly, drawing a direct link between information system utilisation and information systems success.

Focusing in this manner on increased system utilisation as a determinant of success of the information resource has several associated conceptual problems. The use of a computer system can be characterised in many ways, from low level models measuring the number of keystrokes, time on the machine or the number of functions performed (which are further discussed in chapter four), to a higher level measuring the number of reports or letters produced in any given session. Given these multiple characteristics, the choice of measurement device must reflect which aspect of utilisation is to be measured.

Trice and Treacy[45] provide a conceptual framework for utilisation research with 'categorising reference theories'. Four general classes of independent variable emerge: firstly design and implementation process variables that

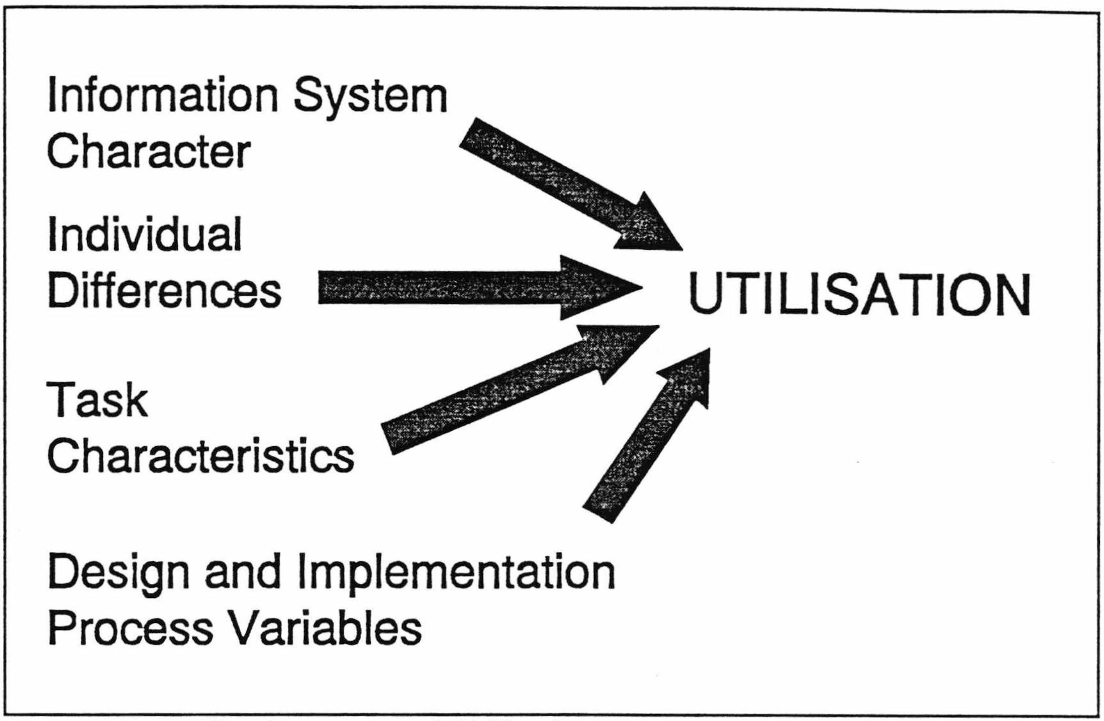

Information System Character
Individual Differences
Task Characteristics
Design and Implementation Process Variables
UTILISATION

Figure 1 The Structure of Utilisation

include training and implementation processes used to install the information system[46]. Here organisation practices have to be unfrozen, examined and refrozen to provide a change mechanism. Secondly information system characteristics. With this group the efficiency and effectiveness of the user physical interaction are considered with an ergonomic orientation, or the development of interface performance and evaluation measures[47,48]. Thirdly task characteristics, the least common area investigated. This area involves looking at the nature of the task to be performed as a measurement of fit to user need. Lastly individual differences based on attitudes, age and education based on theories of reasoned action[49]. The approach taken with this work falls within the characteristics of information systems boundary, focusing on accessibility to the information resource as the link to utilisation and subsequent success of the information resource.

1.4 The Importance of Accessibility

Accessibility to information can be analyised over two levels. Accessing the physical terminal hardware to reach the available machine resources is the first level of accessibility that each user must accomplish. Rice and Shook[50] considered this level of accessibility, particularly with physical terminal location forming a determinant relationship with machine usage patterns. In this context accessibility is based on the physical accessibility to the hardware rather than to the information resource itself.

With increasing availability of hardware through the proliferation of personal computing coupled to networking technology there has been a shift of emphasis away from this hardware accessibility towards what actively mediates between the user and the application. This second aspect of accessibility is increasing in importance as the size and complexity of applications made available to the user community continually increases. Waguespack[51] identifies two layers of complexity: intrinsic and extrinsic. Intrinsic complexity refers to complexity inherent in the application itself, for example a project management application will be inherently more complex than a word processing application. Extrinsic complexity is concerned with the application interface itself, for example the form of dialogue and interface environment provided. With continuing advances in processing power and storage capacity, increasingly large and detailed applications will be generated, for example enabling database modellers to capture growing semantic structures in database applications. Such advances fuel the rising levels of intrinsic complexity to which interfacing software must provide access to a widely diversifying audience. Hence extrinsic complexity is likely to increase, especially in the area of end user database interfacing, ultimately affecting systems and information resource utilisation in the longer term.

O'Reilly[52] investigated the impact of accessibility and perceived information quality on utilisation of the information resource, focusing particularly on why such resources were often left under utilised. O'Reilly argued that the ambiguity of information requested combined with the social and economic costs involved in seeking out optimal information may dominate perceived information quality in determining preferred sources of information. Accessibility to the information resource was established as a key motivation for resource utilisation, those sources of information perceived as difficult to access were left under utilised. However with this study all the available information sources were paper based, none were automated or computerised in any way. With the increasing spread of computer based information resources and diversification of computer staff who come into contact, the question of accessibility has broadened considerably, now including the question of user interface design.

Bozeman and Shangraw[53] examined cost and accessibility issues over a range of both paper and computer based resources for public sector based decision management. Accessibility and cost considerations of information bundles were intertwined in laboratory experiments on volunteer students. Their work established that increasing the 'cost' of any information resource negated accessibility, with subjects tending towards cheaper and more diverse sources of information. On the question of access difficulty, this was found to have only a modest effect: more information resources were consulted as a compensatory measure.

This section has presented the conceptual framework of utilisation, addressing in particular accessibility as an information system characteristic. From the practical viewpoint, accessibility has tended to be mechanistic, an afterthought of the software developer. In practical terms, this question of accessibility in past information systems has been tackled by providing hierarchies of standard reports, forms or menus which were all set up at implementation time, supplemented by a catch all command line query generator. The methodology in providing such a library of reports presupposes all possible decision inquiries are predictable and can be explicitly defined. The real world is rarely this simple, often being composed of rapid unpredictable opportunities demanding by definition unpredictable information requests. With the increasingly diverse user community coming into contact with computer applications and advancing complexity of application design, particularly in the database domain, the role of the interface becomes vital in ensuring accessibility and subsequent utilisation of the application.

1.5 Summary and Research Direction

The role of computer technology within the organisation has filtered up from the shop floor into routine office functions, and now most recently into front line management decision support. Through the process of routinisation and centralisation, computing technology has attempted to maximise performance of both these functions, incurring significant social and organisational change. For both the shop floor and the office, control of accessibility to the technology became a focal point of conflict. This was ultimately resolved by returning accessibility to the end user, for example control of the lathe to the shop floor and control of information processing to the manager's desk.

As computer technology has reached senior management decision support, information systems that were previously perceived as a necessary but undesirable bureaucratic overhead, have now been reassessed in terms of a new economic asset. In recent years, organisations have invested considerable resources in pursuit of perhaps an unquantifiable benefit - improved decision making through increased utilisation of this information resource.

Like any other asset in the organisation, the maintenance and provision of the information resource has to be justified in terms of its effectiveness and efficiency. Using these criteria as a means to judging the success of the information resource is problematic, especially the determination of value based output, for example improved decision making. Several authors have suggested the surrogate measure of utilisation, but this remains a complex variable to evaluate. This work has concentrated on one aspect of information system utilisation that is increasing in importance: user accessibility.

The convergence of increased processing power, mass storage technology and physical accessibility to diversified sources of information has facilitated generation of increasingly large and detailed organisationwide applications. A concurrent effect as the price of this technology has fallen into individual line management budgeting, has been the type of staff that are coming into contact with these computer based applications has diversified, rapidly encompassing non computer literate personnel. These opposing components of utilisation can only intensify as the technology progresses, meeting through the process of user accessibility.

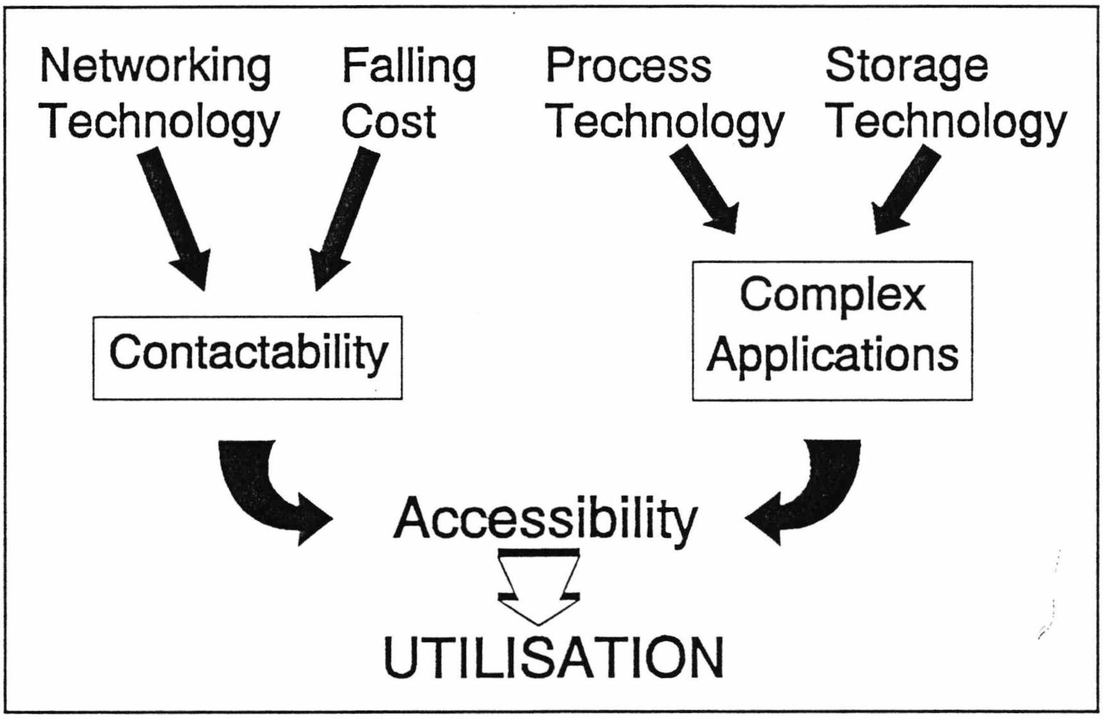


Figure 2 The Accessibility Gap

This research will focus in an applied perspective on this widening gap between non-computer literate personnel and the increasingly complex computer based applications that they are required to manipulate. In particular focusing on the factors that effect utilisation of large information systems in a management setting. For the fieldwork, this management setting was provided in one of the largest organisations in the UK: the National Health Service(NHS). Each component District of the NHS employs an information resource of patient details providing information for local and Regional level decision support. This organisation and particularly this information resource, provided the field study basis to investigate this issue of accessibility and utilisation. To support the field work with some background detail, the next chapter charts the historical development of information within the Service. This can be seen to parallel closely the developments discussed in this chapter, moving from a routinised bureaucratic overhead to the local level information resource of the 1990's. Building on this background perspective, the management study is then presented in chapter three.

CHAPTER TWO

The Changing Role Of Information in the National Health Service

Contents

2.0 Introduction

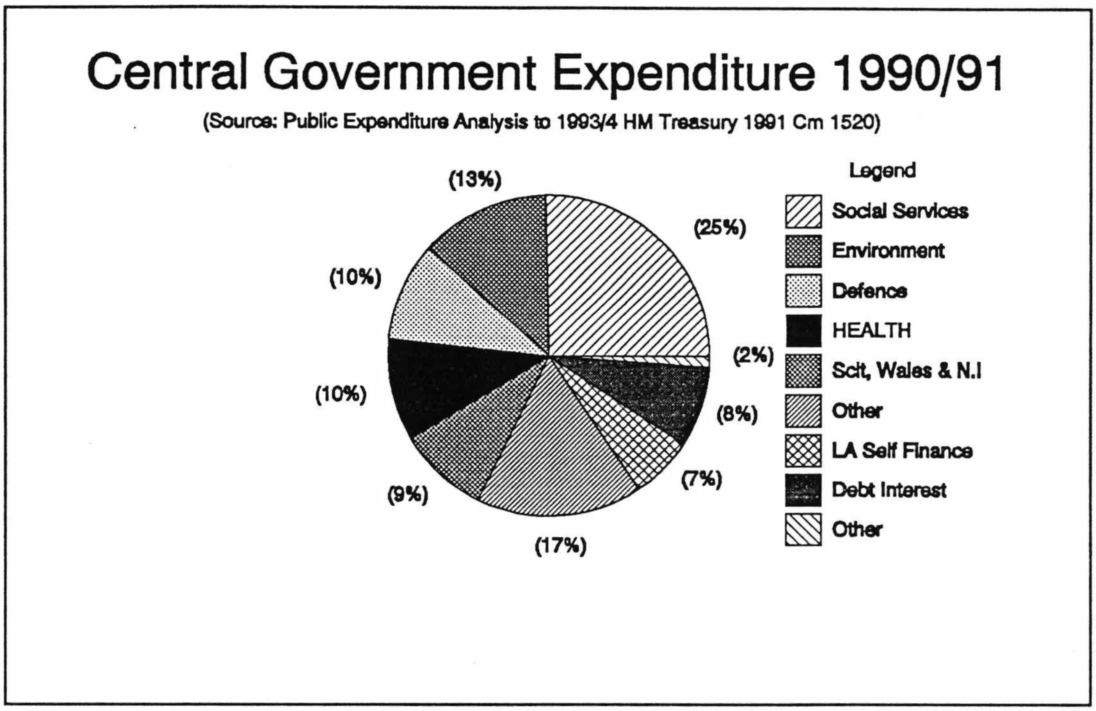
The aim of this chapter is to introduce the background to the work described in Chapter Three. The history of information provision is traced from the creation of the health service through the 1974 and 1982 reorganisations, to the late 1980's and the various information resource management initiatives. The pace of change has been slow, with the National Health Service(NHS) 'evolving' an information system out of multiple independent initiatives. This has mirrored what has been described in general terms in Chapter One: The application of computing technology has progressed from the mechanisation of routine tasks, for example storing patient records through into ad hoc decision support for senior unit level management.

2.1 Background

The NHS in providing medical treatment remains a unique social arrangement in the western world, celebrating its fortieth birthday on 5th July 1988. During this period it has undergone both radical and minor changes primarily brought about via parliamentary intervention. The central pillars of its creation, established by Bevan in 1946: access to comprehensive health care, free of charge on the basis of need, have remained almost intact throughout this period of change, but is questionable whether they will survive so well to its bicentennial anniversary. It is very easy to consider the NHS as a monopoly provider of health care. Although by far the largest, this is not so with other providers including military hospitals, the Public Health Laboratory Service, the Employment Medical Advisory Service, environmental health performed by local government, Home Office Prison Medical Service, the school hospital service, nursing and retirement homes besides the more apparent private health service run solely on the profit motive.

The NHS is a truly massive undertaking, maintaining acute hospital sites, ranging from cottage (although few remain) to extensive district general hospitals(DGH) and the Family Practioner Committees(FPC) overseeing 31,500 (in 1989) general practioners(GPs)[54]. Through this two component system, the NHS provides extensive medical care ranging from maternity, major surgery and Accident and Emergency(A&E) in the acute sector to family medicine and general advice through the local GP. The NHS is funded mainly from direct taxation (97% in 1989), directly employing approximately one million staff[55] making it the second largest, single employer in the Western World (behind the Red Army).

This size of UK health provision has evolved in response to a burgeoning increase in health care demand, based on a zero cost function moulded by social and environmental factors. Examples of some of these factors include social pressure for breast screening, A&E requiring immediate access, location of hospital site - a DGH in the M4 corridor will have a different demand intake to that of Dover DGH, population characteristics - a Southern England coastal DGH will have a markedly different patient age and therefore health care demand when compared to Milton Keynes. Finally new technology not



Central Government Expenditure 1990/91

(Source: Public Expenditure Analysis to 1993/4 HM Treasury 1991 Cm 1520)

Figure 3 Central Government Expenditure 1990/1991

only enables more 'efficient' patient care, but also generates its own demand, for example laser technology has opened up a new outpatient demand for health care previously only available through major surgery.

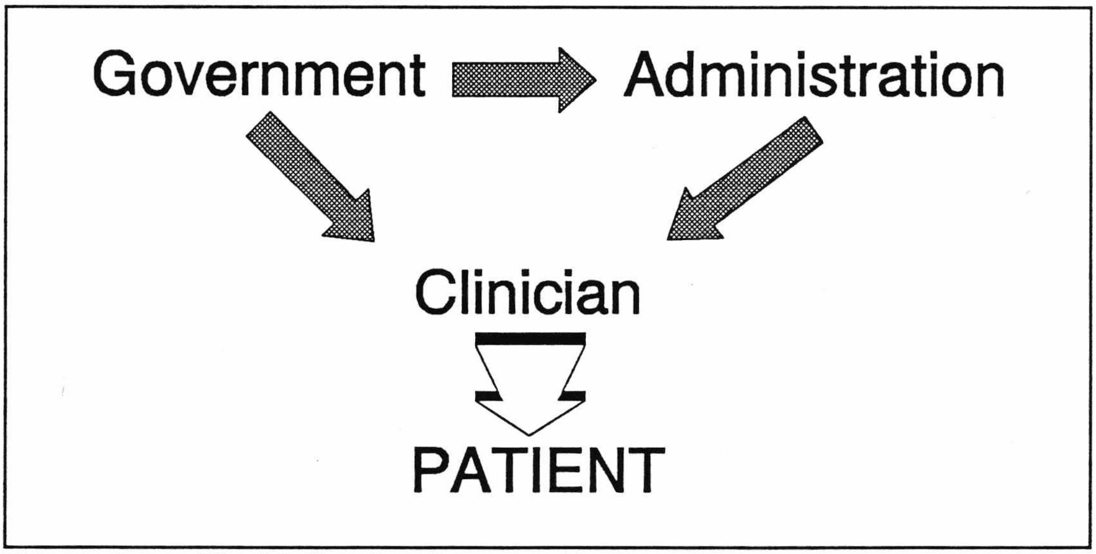


Figure 4 Health Service Triparate Structure

Such a diversity of function, location and patients provide significant problems of control and co-ordination for which the NHS has often been criticised. This is complicated further by the underlying triparate structure of service provision: the integration of government, administration and clinician. Through the life of the Health Service the key problem of any form of management has been one of control. In any commercial organisation this is achieved through the evaluation of input resources compared to productive output. The provision of health care suffers the same problem as education - the measurement of productive output. Input is tangible and readily measured (in total terms however) in monetary units, for example staffing and hotel costs. Output measurement has proved more problematic and has thus tended to be analyised in overall terms for the population as a whole, for example through the use of standardised mortality rates. Limited success has been made with a more theoretical approach of quality adjusted life years, but is difficult to quantify and hence subjective to apply.

Through the 1980s and into the 1990s with an increasing demand for health care, but also with increasing pressure on cost containment, the evaluation of activity as a surrogate for health care effectiveness has been extensively used. As part of this process, the health service has evolved like most other organisations towards a stage of recognising the value of information previously collected on a routine basis. This evolution is described in the remainder of the chapter in four stages of recognition: Health Service beginnings, the first twenty five years, functional management and finally resource management.

2.2 Health Service Beginnings

The establishment of the NHS in 1948 came exactly one hundred years after the amendment to the first Public Health Act and about two hundred years after the inception of the voluntary hospital movement.

As early as 1920 in the Dawson Report[56] produced by the Health Consultative Council on Medical and Allied Services asserted:

> 'the best means of maintaining health and curing disease should be made available to all citizens'.(Para 2.13)

In 1926 the Royal Commission on National Health Insurance concluded,

> 'the ultimate solution will lie, we think, in the direction of divorcing the medical service entirely from the insurance systems and recognising it with all other public health activities as a service to be supported from general public funds' (Para 5.4)

Government participation in health care was extended in 1939 through the Emergency Medical Service reporting to a Minster for Health in Parliament. The principal role of this 'service' was to provide support and centralisation of day to day activities of the various medical facilities available for casualties during the war effort.

In 1942 the Beveridge Report[57] made far reaching recommendations for the comprehensive provision of health care to all citizens These assertions formed the basis of postwar health care in the UK. The coalition government of 1943 agreed to the concept of a national unified comprehensive health scheme, raising health care to a universal right to all citizens.

Considerable debate continued culminating with publication of the White Paper entitled 'A National Health Service'[58] in 1944 presenting the notion of a unified service. Subsequently Aneurin Bevan's 'The NHS Bill'[59] in 1946 was published, with the appointed day set to 5th July 1948. The 1946 Act provided for:

> 'the establishment in England and Wales of a comprehensive health service designed to secure improvement in the physical and mental health of the people of England and Wales and the prevention, diagnosis and treatment of illness' (Part 1 Section 1)

The primary concentration was on the establishment of a unified service giving a universal right to health care,

> 'the biggest single advance ever made in this country in the sphere of public health'[60]

The NHS of July 1948 came about as an amalgamation of three previously distinct areas of health care provision:

- the voluntary hospital service
- the public health service
- the mental institutions

The voluntary health service provided 1143 hospitals with 90,000 beds in old prestigious hospitals with heavy administration and accelerating costs. The public health service as established from the Local Government Act 1923[61] and the Poor Law Act (Amendment) 1934[62] varied considerably in quality and scale of sites, providing 1545 municipal hospitals with 390,000 beds. Mental health under local authority control from the Lunacy Act 1890[63] provided another 190,000 beds, totalling 670,000 beds.

To manage this spread of health care, the Act provided for a triparate management structure:

- Hospital, Specialist and Ancillary Services
- Family Practioner Services and
- Local Health Services.

For Hospital, Specialist and Ancillary Services, the country was split into fourteen regional health boards(RHB) each centred around a University medical institution. Hospital management committees were appointed to run non teaching hospitals - 388 in total, and Boards of Governors to run the 36 teaching hospitals directly. Family Practioner Committees were established as a semi-autonomous body to oversee GP medical provision. The Local Health Services were maintained by local authorities which through their Health Committees provided environmental health and community services, for example maternity aftercare and health centre maintenance.

2.3 The First Years:

Information reporting during the first stages of Health Service development was patchy in quality, concentrating in particular on expenditure inputs. Basic resource monitoring was carried out under Statutory Instrument 1414 issued in 1906, built around 'subjective costing' returns to the Ministry of Health as part of the annual Treasury costing cycle. Subjective costing entailed the allocation of resources to broad cost categories, with spending within these limits delegated to the management concerned. The Treasury only monitored total spending with any savings clawed back to fund other overspending departments.

The evaluation of 'output' from the Health Service was limited to ad hoc reporting directly to parliament. These were normally contracted to agencies external to the NHS, such as the General Register Office, subsequently the Office of Population, Census and Surveys(OPCS) and later the World Health Organisation (WHO). These comprised generalised indicators of how the overall health of the nation was changing. Parliament had therefore very little control or feedback over the activities of the health service, acting only as a financier with no real resource utilisation data.

Demand for health care continued to rise rapidly, constantly outstripping supply. The total cost (in 1948/49 prices) had risen by £73.2 million to £406.4 million by 1953/54. Beveridge's white paper 'A National Health Service' estimated £130 million for the first year, falling past a peak into cost savings as the total health of the nation improved. This was never realised, with expenditure growth on the Health Service increasing ever since.

The election of February 1950 returned a Labour government with a mandate to investigate this increasing cost. By April 1951 the NHS Amendment Bill[64] was passed, which for the first time involved limited resource allocation, pegged at £392 million. In addition fees were introduced, covering such items as dentures, wigs, glasses and drugs supplied to outpatients, and the supply,

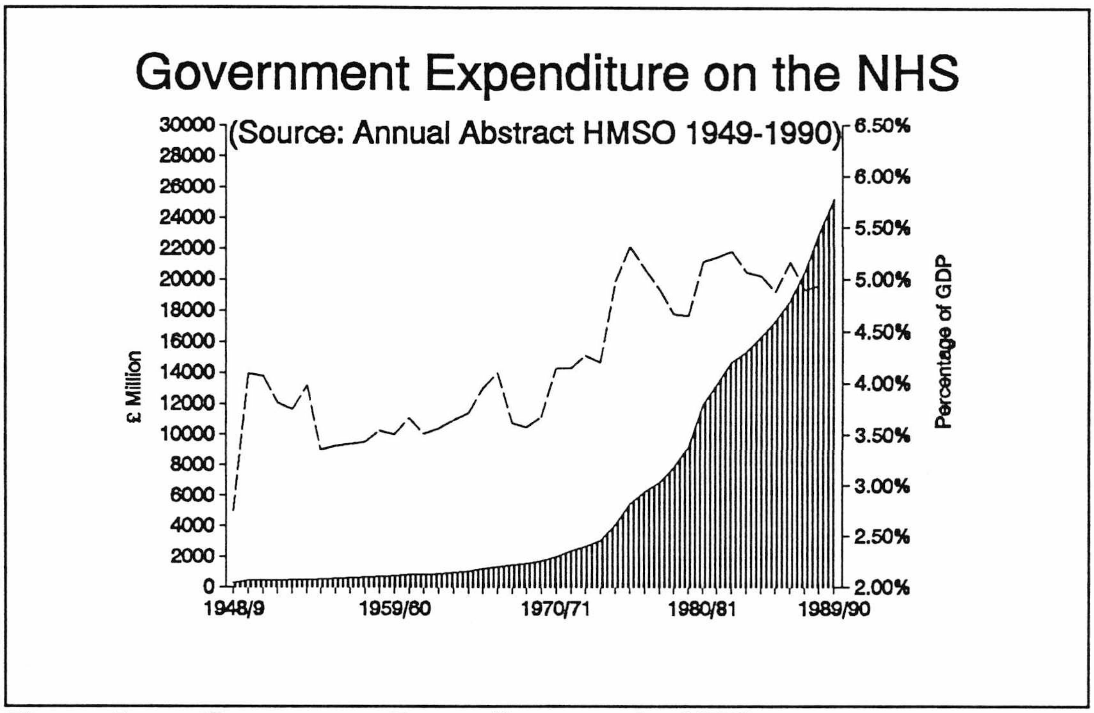


Figure 5 Government Expenditure on the NHS

repair or replacement of surgical supports. Such movement ended the concept of a comprehensive, free Health Service at the point of access. With this action there was an attempt to restrain expenditure growth of the NHS, which remains to date a key annual process of Health Service finance.

From this period several ways were developed to increase accountability of the growing Service to Parliament. In 1953 a Committee of Enquiry into the Cost of the NHS (the Guillebaud Committee)[65] was set up recommending in November 1955 three approaches to better management of the cost function. Principally each hospital authority was to send a monthly return (called a Monthly Hospital Return) to the Ministry of Health

> 'showing how expenditure to end of the proceeding month compared with due proportion of apportioned estimates'(Para 367)

In addition expenditure was brought under budgeting heads with underspends reallocated to overspending heads. To provide a direct reporting mechanism, a Regional Officer was established to observe key health area meetings, reporting directly to the Ministry.

Essentially a movement away from subjective costing towards a system of functional budgetary control at hospital and department levels was recommended, 'setting standards of efficiency each year'(Para 365) against which actual spending was to be compared through the Service. At this time there was no central form of statistical or information function. Resulting from the severe criticism made by this committee:

> 'it is a matter of regret that for the first seven years of the Service, the Ministry should have been without the services of a qualified statistician' (Para 364)

in July 1955 the first Ministry Statistician was appointed. During this same period the Ministry had established a Hospital Organisation and Methods Service(HOMS) providing advice and consultation in general management to any department. This focused however on working practices rather than information generation. Through this report, the Ministry established a Research and Statistics Department (HM(54)64) to compliment the HOMS department, devoting all of its time to generation of statistical information. This report was the first to identify the importance of information and the clinicians role in control of resources and information provision. For example 'doctors should be aware of the cost of what they are doing and have responsibility' (Para 699)

This Research and Statistics department was kept purposefully small, targeting only overall health statistics, for example bed occupancy measured at midnight. External NHS agencies were still contracted for ad hoc parliamentary reporting. From these routinised statistics, OPCS generated the Hospital Input Enquiry(HIPE), running from 1957 to replacement by Korner returns in 1982. Based on a simple one in ten sample of all acute patients, this provided overall statistical planning information for service provision to the Ministry, for example hospital load and throughput. It was not until 1967 and the First Report of the Joint Working Board on Medical Work in Hospitals (the Cogwheel Reports)[66] that information and local level planning needs were fully appreciated. The centralised system of data collection specifically for centralised use was identified as inadequate:

> 'statistical and other sources of information now delineate the community's need for indepth and detail never possible before. This sharper perception of the problems of planning and of evaluation of medical care services demands a higher level of efficiency than that of which existing machinery is capable' (Para 73)

The report recommended setting up Hospital Activity Returns based upon the 1955 Hospital Statistical Returns but targeted at local level or 'unit' management. These were first published in 1969, with data grouped around 'specialities' standardised for comparability. This reporting structure remained intact until replacement by Korner returns in 1982.

During this initial rapid period of growth, concern was repeatedly expressed for the manageability of such a diverse organisation. Parliament had focused on the budgetary measure as a resource allocation mechanism, with only limited reporting on the effectiveness of the health care delivery. It was not until 1956 that the statistical function was recognised, providing the beginning of standardised data reporting. The Guillebaud report identified the need for local level information provision, which was reaffirmed by the Cogwheel reports, identifying a 'unit' level need for information provision. This formed the beginnings of a shift away from centralised, towards local level information provision.

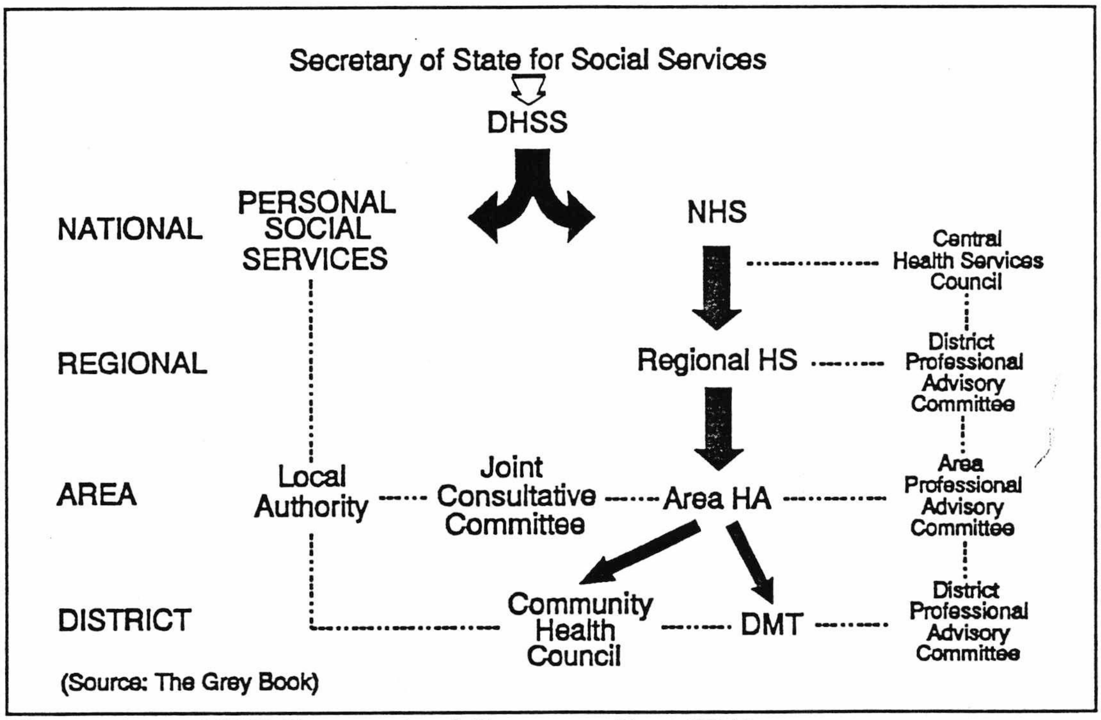
2.4 Functional Management

In 1974 the Health Service underwent its first significant reorganisation. The triparate structure of hospital services, local health services and executive councils had remained unchanged since the inception of the health service in 1948. This was increasingly questioned towards the end of the 1960s, for example by the Doctors themselves through the Porrit report[67]. This culminated in a statement to the House Of Commons by Kenneth Robinson, Labour Health Minister announcing a

> 'full and careful examination of the administrative structure of the Health Service'[68].

Two green papers, 'The Administration Structure of Medical and Related Services in England and Wales'[69] and 'The Future Structure of the National Health Service[70]' together with a White Paper on local government in England[71] were published, reflecting the depth and differences in discussion over possible restructuring. There was for example, a strong lobby for returning administration to the local authority level. The final stages to reorganisation were laid by the White Paper 'National Health Service: Reorganisation in England'[72] in August 1972 and the 'National Health Service Reorganisation Bill' in November 1972, arriving finally at the 'National Health Service Reorganisation Act'[73] in 1972. This received royal assent exactly twenty five years after the 'appointed day' on 5th July 1972.

The management arrangements for the reorganisation, known as the Grey Book[74] appeared towards the end of 1972. This provided the skeleton structure of the new organisation, in particular specification of Regional Health Authority, Area Health Authority, and District Management Teams(DMT).


Secretary of State for Social Services
DHSS
NATIONAL
PERSONAL SOCIAL SERVICES
NHS
Central Health Services Council
REGIONAL
Regional HS
District Professional Advisory Committee
AREA
Local Authority
Joint Consultative Committee
Area HA
Area Professional Advisory Committee
DISTRICT
Community Health Council
DMT
District Professional Advisory Committee

(Source: The Grey Book)

Figure 6 NHS Reorganised Structure Post 1974

The aim was to couple clear responsibility with maximum decentralisation.

The definition of accountability was recognised as problematic because consultants and GP's are

> 'primarily accountable to their patients'(Para 1.18)

This conflict of interest was tackled through the development of plans for each level of the organisation as a

> 'yardstick against which to measure performance and which will contain targets as incentives' (Para 2.32)

It was recognised that information is available at all levels in the organisation, forming the basis of these plans, but is

> 'sometimes unreliable, of doubtful relevance and out of date, and there are gaps in what is available'(Para 2.34)

To address this problem, a systematic assessment of what information is needed at each management level and the establishment of an expert information consultancy function at Area Level was recommended. The role of the clinician in determining the demand for resources was identified but taken no further.

This was the first attempt at an explicit formalised planning system that recognised the importance of information at *all* levels of the Health Service. Monitoring of the Service by Parliament was again left on a cost input basis - the expenditure budget. The role of the clinician in the management process was likewise identified but left unaddressed.

On the very day the reorganised health service came into effect (5th July 1974), plans were being made for further change. In February 1974 a Labour government was elected with Barbara Castle as Health Minister. The inherited structure devised by Sir Keith Josepth continued to be implemented, but in May 1976 Barbara Castle invited Sir Alex Merrison, Vice Chancellor of Bristol University, to chair an investigating Royal Commission[75] with the following terms of reference:

> 'To consider in the interests of both the patients and of those who work in the NHS the best use and management of the financial and manpower resources of the NHS'(Para 3).

This provided the widest and most comprehensive review of the Health Service since the Guillebaud Report in 1956. Three years and £918,000 were invested, generating a wide range of recommendations, concluding ultimately that

> 'we need not be ashamed of our Health Service and that there are many aspects of it which we can be justly proud' (Para 2.23)

The report in general was critical of consensus management, identifying for the first time in a formal manner the need for clinicians involvement as 'explicit resource managers'(Para 21.55) in the management of the health service. The report recommended experiments and encouragement of clinical budgeting, whilst also recognising the significant, associated practical difficulties. Information was identified as a key ingredient, enabling management to make the best use of available resources. However it was also recognised that

> 'Rational decisions on priorities are impossible with judgements of efficiency of service resting on insecure foundations' (para 21.57)

The insecure foundations referred to the information systems found in the Service:

> 'information available to assist decision makers in the NHS leaves much to be desired. Relevant information may not be available at all, or in the wrong form. Information that is produced is often too late to assist decisions and may be of dubious accuracy.'(para 21.56)

Merrison, drawing on Perrin's report[76] identified a total lack of costing information. No inventory of capital assets was available and subsequently no form of patient costing could be performed. The current system of financial management did not encourage efficient resource usage (para 22.76). The report identified a need for increased expenditure on administration to generate an expected increase in the quality of decision making (para 21.60). No attempt was made to quantify either.

Despite the problems identified in measuring efficiency in health care, the report focused on making better use of the resources available to the Health Service (para 22.11) through improved decision making. The report concluded:

> 'much of the information required for effective management was not produced or was inaccurate or too late to be of value' para 22.76

Through the early 1970's, health service spending increased significantly, rising from £1705 millions in 1970 to £4095 millions in 1974. A Resource Allocation Working Party(RAWP) was set up, reporting in 1976[77] as part of the aim to redistribute resources between health authorities according to level of mortality. In effect this meant moving resources away from the bias towards London, inherited from the triparate structure in 1948 as reinforced through nearly three decades of incremental budgeting. Through the imposition of incremental processes of functional budgeting, the government extended lines of accountability, focusing on controlling resource input. Output measurements relied on top down HAA and HIPE returns, supplemented by outside reporting bodies. These measurements remained of little value to local level management.

Through this period there was growing awareness that information for resource allocation and health care output was lacking. Principally there was a realisation that the clinician should be brought into the budgeting process, as consumption of resources remained under the control of the clinician with primary responsibility to the patient. The budgetary responsibility of the clinicians actions remained separate, lying with the associated line management.

2.5 Resource Management

In May 1979 a Conservative government was elected, inheriting the Merrison report not of its own making, a situation similar to that encountered by the Labour government and the Guillebaud Report of 1956. In December 1979 the government's response, 'Patients First'[78] was published and subsequently implemented through HC(80)8 'Health Service: Structure and Management'[79]. This provided for a second reorganisation, wary of the costs of the first, on a smaller, faster scale towards a deadline of 1st April 1982. As a starting point, Patients First noted

> 'the organisation did not provide the best framework for the effective delivery of care to patients'(Forward Para 1).

The report concentrated on delegation of authority with decisions made closer to the patient. District Health Authorities were identified as the focus of management action, moving away from the functional organisation of the past thirty years. The Area tier of organisation was to be removed, streamlining management into Regional(RHA) and District(DHA) Health authorities.

With this change in structural emphasis, the information needs of the Service changed. Previously information systems were built around functional cost management, now with site or unit management autonomy, information systems were to integrate input (cost) and activity information, for example through the realigning of financial with activity reporting years.

HN(79)21 Information Requirements of the Health Service[80] noted the concentration on input information and almost total disregard for detailed output information in the functional organisation. In particular the health notice was 'critical in tenor' (Para 7) of information provision in the Service. Data was provided top down, focused primarily on the needs of the department(DHSS) rather than the NHS as a whole. (Para 4) Significant amounts of information were collected and reported, but much was seen as unreliable, inaccurate or incomplete (Paras 8 and 9).

Following this in February 1980, the DHSS announced the formation of a DHSS/NHS Steering Group on Health Service Information[81]. This was chaired by Mrs Edith Korner with the following terms of reference:

> 1/ to agree, implement and keep under review principles and procedures to guide the future development of health services information systems
> 2/ to identify and resolve health services information issues requiring a coordinated approach
> 3/ to review existing health services information systems and
> 4/ to consider proposals for changes to, or developments in, health services information systems arising elsewhere and if acceptable, to assess priorities for their development and implementation'(Para 1.1)

In more general terms the aim was to

> 'assist in the creation of an environment which will allow and encourage the efficient collection, processing and transmission of data. We therefore see it as part of our task to....promote the use of Information Technology and stimulate the development of computer systems to enable the integration, ease of transmission and distribution of access to data'(Para 1.13)

This was the first time that information provision within the Health Service was analyised in a practical overview of activity, for example covering patient related activities such as ambulance transport, financial services as well as patient information itself. The reports were by no means exhaustive however, ignoring areas such as theatre administration, epidemiology, purchasing and stores. Six reports were produced from May 1982 to May 1984 covering:

- Hospital and Diagnostic Activity
- Patient Transport Services
- Hospital Services Manpower
- Paramedical And Miscellaneous
- Community Services
- Finance

The key contribution from these reports was the identification of minimum data sets at critical moments through a patients interaction with the Health Service. Such information was where possible, determined as a byproduct to operational procedures focused at DHA level. The First Report recommends the collection of a minimum set of data

> 'by a single patient information system on all patients who occupy a bed in a ward'(Para 3.17)

generating ward stay and patient episode data at the District level. In effect, Korner concentrated on clinical services carried out to patients, rather than on an exhaustive patient based statistics reporting system.

After initial discussions of the various methods of patient care analysis, for example: patient costing, diagnostic related costing, clinical team costing and client group costing, the Sixth Korner Report, followed the initial suggestions from the Cogwheel Reports: cost statements were to be produced based on speciality, that is on groupings of patient complaints:

> 'We believe that the absence of a uniform, generally available cost analysis of different kinds of patient care in hospital and community services by patient group characteristics (eg age, diagnosis, speciality) severely limits the ability of districts to manage, and adversely affects planning, monitoring and performance evaluation at all levels.' (Para 2.23) 'We therefore recommend that speciality costing be introduced.' (Para 2.27)

Speciality costing was to be viewed as an intermediary stage towards the more flexible but not yet attainable patient costing. The focus of information gathering and reporting remained as in all previous reports, built around input data, effectively replacing HAA and HIPE.(SM(85)2/14) No guidance was given for the collection or generation of output information.

All six reports were accepted and implemented through HC(84)10. These Korner reports provided the basis of information flows within the NHS. The beginnings of an information culture was established, upon which all subsequent information developments have been built. Development of the Corporate NHS Data Model, launched in 1989 and revised May 1991, and the Computer Policy Committee, flowed directly from the first Korner report on Health Service Information.

Through increasing government insistence on efficiency, autonomy of unit management was strengthened with budgetary and accounting review process implementation. As part of this, the government looked to the Service itself to provide individual output indicators, rather than on behalf of the Service as a whole via changing population statistics. A revised planning system was introduced in 1982 (HC(82)6)[82] and systems of annual accountability review established in HC(82)2[83]. The aim of this notice was to provide better accountability of the NHS to government, through annual Regional Review Meetings. These would be led by the Minister responsible to parliament for the NHS, investigating long terms plans, objectives and the overall effectiveness of each Region. The Minister in particular would require each RHA to provide evidence that government policies were being followed and agree objectives for the coming year. This accounting review process was then extended through to District level (HC(82)14) and subsequently unit level in 1983 to promote accountability into the lower structure levels. On this basis the planning systems for the NHS was revised to a more business like footing

(HC(82)6) with each tier required to produce a ten year Strategic Plan on a five yearly basis. Other planning issues were tackled, for example Manpower (the primary NHS cost) information was provided more quickly and at quarterly intervals in an attempt to tighten manpower planning and control.

As a specific attempt to evaluate health care outcomes, an important step was made in linking cost input to service activity through the development of approximately 145 performance indicators(PI)(HN(83)25[84], HC(85)23[85] and HN(85)32)[86]. In July 1983 the Joint Group on Performance Indicators was established to consider future development and use of performance indicators. This formed the basis of HN(86)36 requiring all Districts to produce the required PI's. Based on a BBC B micro computer, this computer package brought together for the first time data from finance, activity analysis (principally from HAA and HIPE but later standardised Korner returns) and manpower in a form not previously possible.

It was widely recognised, especially by critics of the DHSS, that the information derived from these initiatives was only as good as the feeder input systems employed through each District. During this period these feeder systems were in a considerable state of turmoil, either in the process of original implementation or in a state of transition to standardised Korner returns. Information for measuring performance from PI's was widely recognised therefore as crude and of little practical use. This was however not the key objective. The key aim of these initiatives was to build on the information culture of Korner, introducing clinicians to the idea that their decisions translate directly into resource consumption considerations. For the first time an attempt was made to bridge this gap.

The government was pursuing the drive for efficiency in other ways, in particular by opening up health care provision to market forces. This took the form of competitive tendering introduced by HC(83)14 NHS Support Services: Contracting Out[87], HC(83)18 'Competitive Tendering In The Provision of Domestic, Catering and Laundry Services'[88] and DA(83)40 Competitive Tendering In The Provision of Domestic and Laundry Services (Specimen Contracts)[89] on the cost side and tax incentives for private medicine aiming to increase consumer choice.

As part of this drive, Mr Roy Griffiths Chairman of Sainsbury's was appointed to Chair a Committee of Inquiry Into The Management of the NHS that reported in October 1982[90]. Griffiths identified a lack of drive, primarily residual from the DMT structure of decision making with no one person responsible for activity. Consensus management had led to delays, often resulting in lowest common denominator management:

> 'In short, if Florence Nightingale were carrying her lamp through the corridors of the NHS today, she would almost certainly be searching for the people in charge' (Para 5)

Griffiths made far reaching recommendations targeted at initiating major changes in NHS management culture. An Executive Committee was installed at national and Regional levels (NHS Management Board and Health Service Supervisory Board) and executive members at both District and Unit level as focal points for authority (HC(84)13[91]). Lines of accountability were strengthened, providing a leadership function at all levels. Both management by consensus and functional management were finally replaced with 'general management'(para 7). The aim was to provide unit management with 'strictly relevant management information'(para 8.2), determined at the local level(para 6) in the development of unit level management budgets. In particular they were to involve clinicians in this process, relating workload and service objectives to financial and manpower allocations.

Griffiths tried to integrate the service provider (clinician) and budget holder, thereby bringing control of resource consumption into the budgeting process and hence indirect parliamentary control. The doctor is the natural manager of the economic resources available to the NHS (para 19), and so should be the budget holder. The Griffiths proposals, together with other government initiatives that followed, for example the White Paper 'Working For Patients' rest upon the provision of integrated, up to date information. As a report from the Nuffield Institute[92] recognised:

> 'for a variety of reasons including lack of manpower, lack of skills, lack of investment, lack of properly designed administration pathways and capabilities, most of our information services were established piecemeal and directed towards particular problems rather than of components of a well thought out general strategy' (p3)

To provide a basis for these government initiatives, some form of integration of the diverse information systems to be found in the Service was required.

HC(86)34[93] announced the beginning of a Resource Management Initiative(RMI). This recognised the problems of diverse sources of information required for unit level budgeting and eventually patient care and cost planning. Six pilot studies were set up based on those Districts that showed a 'progressive attitude', for example Newcastle's Freeman Hospital for the development of case mix management information system(CMMIS). These systems were structured to draw information from several sources, for example PAS, radiology, X Ray and pathology into an integrated case mix database, containing records of individual episodes of care and provision of means to attach costs to them. The success of these initiatives is difficult to quantify, as the report by Brunel Health Economic Research Group[94] identified:

> 'the quality of available data on the overall performance of the six sites relative to the national picture is not good enough to make firm comparisons'(Para 6.8)

The 'rolling out' of the RMI towards the marginal 'progressive' computer driven Districts will intensify this evaluation problem, giving rise to further allegations of inappropriate resource allocation within the Service.

2.6 Information Management and The Future

The basis for measuring activity associated with each patient or 'critical moments' was established through the Korner Reports, with the provision of computerised patient administration systems at District level. The integration of this patient activity information into a wider information system of financial and manpower allocations provides scope for the development of patient cost profiles or patient 'care plans' and subsequent inter-District charging mechanisms. The potential arises in the future therefore, for charging of cross boundary patients and the development of Trusts Status for those Districts who wish to opt out of Regional supervision. This was the theme of the White Paper 'Working For Patients'[95] which is to instal a buyer/provider split between units, hospitals and the District. To effect such a relationship, the focus of information provision has shifted from historical analysis to timely up to date information for current interpretation, for example contract negotiation and particularly outcome evaluation. Such a change in emphasis for information provision is forcing Districts to reassess their information systems towards a more commercial perspective. In particular

> 'Interaction and communication between information systems are not just becoming desirable, they are becoming indispensable if new management systems, process and functions are too reach the levels of sophistication and efficiency which the NHS must achieve in a modern world.'(Para 31.1)[96]

Information in the Health Service has evolved, paralleling the organisation of the Service itself. Parliament has principally been concerned with costing and resource information based around subjective costing principles, continuing through to formal budgeting and lines of accountability in the 1974 re-organisation. As the cost of health care provision has increased in all industrial countries, various governments have demanded better 'value for money' health care provision. The UK was well placed with the 1982 reorganisation moving the budgeting process closer to the patient - to hospital and unit level. Despite these various initiatives, control of resource consumption remained divorced from budgetary control. Griffiths began a process of bringing the clinician directly into this budgeting process as a key mechanism to control cost activity.

The focus of each initiative has been towards developing systems to monitor activity. Little work has been directed towards outcome measurement, that is activity is taken as a proxy for effective treatment. Through the early 1950/60's it was comparatively simple to measure Service overall effectiveness via changes in the overall population health, for example through a falling mortality rate. This approach provides little evaluation through time, being complicated by social and environmental factors, for example new technology. In particular governments have sought ways of evaluating individual activity within the service as a means of cost containment. One approach suggested in the late 1960's addressing this issue was the use of quality adjusted life years(QALY) of a patient after treatment. This approach has many moral and conceptual problems much discussed in the damages and compensation sphere of the legal profession, for example in cases such as Kelly et al vs Dawes[97] and Burke vs London Borough of Tower Hamlets[98].

In practical terms the initial point of contact for most patients, and all follow up treatments is via a family doctor(GP). However because of the relatively small costs involved, 90% of contacts amounting to only 10% of overall NHS budgeted expenditure, parliament has concentrated on the acute medical sector of health provision. In recent years both GP's themselves and the Family Practioner Committee(FPC) have developed their own computer systems.

These remain in isolation of hospital information systems even though they are intrinsically linked, storing information about the same patient. With increasing integration of acute information systems and the falling cost of networking hardware, there exists great potential to provide GP's with direct access to hospital based information. This will undoubtedly include patient based information, but with the integration of financial and activity information, treatment cost profiles will be available. This will enable GP's to refer patients to the 'cheapest' or 'fastest' service provider, especially as GP's are coming under budgetary control themselves. This integration of information systems within the Health Service is recognised as having great potential both in terms of integration as a whole but also as a tool enabling cost efficiencies to be applied, measured regardless of political model. This cannot be avoided as the NHS has concerned Parliament since its very inception in the 1940's and is likely to remain so, commanding as it does 4.9%GDP (1990) of national resource consumption.

This chapter has followed the National Health Service through an information evolution, from uncultivated information provision, arriving in the 1990's with information as a corporate resource, requiring effective management and local level delivery. As the Service comes under increasing resource constraint, key resource questions are raised with each new capital investment that could otherwise be channelled into 'improved patient care'. The benefits of such information technology investment remain intangible and difficult to define in real terms. The key indicator of information system success, as discussed in

Chapter One is that of utilisation as a surrogate measure for effectiveness. The next chapter will present field work research into this area concentrating on the provision of patient based information for local level ad hoc decision making.

CHAPTER THREE

Fieldwork Study with the National Health Service

Contents

3.1 Introduction

The role of computing within the organisation is changing. Information systems have often evolved in a piecemeal fashion, in response to localised need and direction. With falling cost, increasing capacity of processing power, and networking technology, information and its provision has been reassessed as a new form of organisational asset.

Chapter Two traced information provision through the historical development of health care provision in the UK, moving from a centralised bureaucratic overhead to a local level management tool. Significant resources have been invested in this on going transition. To make the transition complete however, further resources will be required, the value of which are becoming increasingly difficult to justify. For example the Service was founded in 1948 on criteria such as equality of access, comprehensive service delivery and free of expense at point of access. Through the workings of time and successive governments, additional criteria have been added - those of efficiency and effectiveness. Chapter One identified and discussed the problems of such an analysis with regard to information systems.

For the Health Service in particular, there is an additional problem: as a government and therefore politically funded service functioning at zero cost, demand is effectively infinite but resource allocation finite. Many observers both within and outside the Health Service see investment in IT as a diversion of funds away from patient services and ultimately 'patient care'. Such allegations are often expressed in the media. It is rarely possible to identify an immediate payoff for such investment, but correspondingly simple to quantify the increased costs involved, for example a short term increase in staffing (the key cost of the NHS). This whole issue of questioning the success of information technology investment or success of the information resource, is often raised. For the future though, this can only recur more frequently as the NHS comes under increasing cost constraint.

Chapter One identified utilisation as a surrogate measure for information systems success. The Health Service has many information systems, both manual and computerised. This study has aimed to concentrate on one key system which forms the cornerstone to all other information systems: the patient administration system(PAS). The aim of this study has been to investigate in practical terms those factors which affect its utilisation and hence organisational 'success'.

As a result of the Korner initiative during the early 1980's, a patient administration system was to be installed in each District Health Authority (HC(84)10). The aim of such a system was to capture all patient related activity through critical moments of patient/Health Service interaction. PAS was to be a standardised information resource against which users from different aspects of the Health Service could draw information: consultants for patient based information, line or unit management for local level service load and throughput information, and senior management/Region for strategy and planning issues. The Korner committee only provided specification of the input function to such an information system, output specification was left unaddressed, leaving Districts to develop individual patterns of utilisation.

Through Inter Regional Collaboration(IRC) several Regions took on standardised hard and software from a single supplier. Participating Regions then imposed this standardised approach on each constituent District. By focusing on a Region which participated in this IRC initiative, many external variables which would otherwise account for significant variation in utilisation are standardised or controlled out of the study. For example hardware provision, data modelling structures used, querying language and interface design were all standardised across participating Districts.

The Region chosen in this study was a member of the IRC, providing a Region wide implementation based on ICL hard and software. ICL provided a product called DIS(District Information System) based on a networked IDMS database implementation of the standardised NHS Data Model for inpatient enquiry. This is coupled to PAS via a daily update mounted on the same hardware[1]. All Districts were directed by Region, that by Spring 1987 this product should be installed and operational in time to meet the Korner returns deadline of April 1987. By Summer 1987, the last District had a DIS implementation operational. This study was conducted through the summer

[1] Further background description of PAS and DIS and their interrelationship is provided in Appendix One.

of 1989, investigating differing patterns of resource utilisation which have matured over the two years the systems have been operational.

3.2 Research Design

This work has comprised of two stages: a pilot and a main study.

3.2.1 Stage One: The Pilot Study

The initial stage of the project was to carry out a pilot study. This had two aims. Firstly an assessment was required of those factors relating to utilisation which required further investigation. Secondly on completion of the first aim, both interview format and questionnaire construction were to be finalised. Two pilot studies were completed, one with a local District Health Authority, the other with the county Social Services Department. The Social Services Department used the same ICL hardware and ICL IDMS networked database construction in a similar application domain to the health authority pilot. Interviews were carried out in the health authority with the District Information Manager and IT management in general, supplemented with on-site observations at all levels, from machine room to senior management. The Social Services study provided an opportunity to observe a different organisation using a similar information infrastructure. Interviews and discussion were carried out with IT management, programming and information support staff which were again supplemented by on-site observations. These initial studies provided a valuable insight into areas to be addressed in more detail in the main study, particularly interview topics to be covered and questionnaire design.

3.2.2 Stage Two: Main Study

3.2.2.1 Coverage

The host Region had fifteen Districts, two of which contain London teaching hospitals. A sample was chosen based on these variables as selection criteria: District size (including two greater London sites), progress with Korner returns (as established from the pilot study) and Resource Allocation Working Party (RAWP) formula allocation. Nine out of the fifteen were approached of which seven accepted, giving a District coverage of 46% or 53% of Regional population. Of these seven, four were RAWP gainers, one was relatively static and two were losers (the London sites).

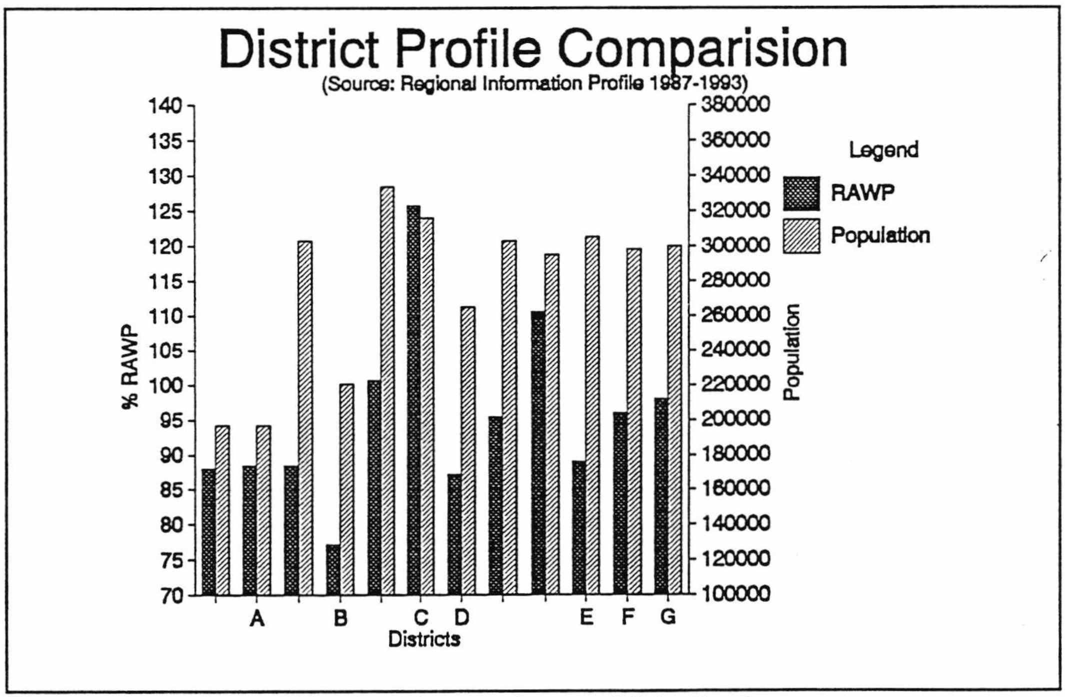


Figure 7 Inter District Comparison within Region

Each Information Manager was approached via an informal interview and an individually tailored programme drawn up based on interviewing key staff who come into contact with PAS or DIS. The work has involved a combination of structured open discussions, observation work and each member of staff completing a confidential questionnaire.

For the purpose of this thesis, each District will be referred to by letter (allocated at random) to maintain confidentiality which was assured to all participants in the study. A summary of each District characteristics is provided here, further detailed profiles are provided in Appendix Two.

3.2.2.2 Study Area Profiles

The Region is composed of 125 sites covering 1474 hectares of notional value greater than £1300 million. The land area is split into fifteen Districts with half of all hospital sites constructed before 1914 and two thirds before 1939. The total population has been comparatively static since 1978 at 3.6 million, but has experienced a significant shift in age structure. This Region provides standard accounting and stock management systems, child health care systems and AGIS standard activity analysis for all constituent Districts.

District A: Single DGH serving 196,900 population. RAWP gainer 88.5% to 95% by 1993. Information function employs 5 staff responsible to the Director of Information Services/Resource Management.

District B: Large London teaching DGH serving falling population of 162,800 but has national specialism catchment. RAWP loser 149.5% to 121.8% by 1993. Own home grown PAS installed for some time. Good training facilities providing an Information shop plus regular Information Bulletins. Three staff are employed in the information function, reporting to the Director of Public Health.

District C: Bi polar DGH District serving 303,000 population with population in the 85+ age group projected to increase by 40% by 1993. RAWP gainer 88.5% to 95% in 1993. Two people provide the information function responsible to the Director of Planning and Information Management.

District D: Bi polar DGH District serving 220,500 population. RAWP static at 100.7% Combined Information and Information Technology function employing four staff plus a Clinical Information Officer with direct responsibility to the Director of Finance and Information.

District E: Single large modern DGH (which is being extended) serving 333,800 population. RAWP gainer 77.2% to 95% by 1993. Four staff employed in information function responsible to the Director of Public Health.

District F: Bi polar DGH District serving 316,000 population plus national specialism catchment. RAWP loser 125.7% to 109.8% by 1993. Organisation split into management units with information function attached to units as required. In addition maintain a central Information Department responsible to the Director of Public Health.

District G Bi polar DGH coastal District serving 265,000 base population which varies considerably through the year. RAWP gainer 87.2% to 95% in 1993. Small combined Information and Computing function employing two staff.

3.2.2.3 Interview Methodology

Research which involves interviewing is often based around a rigid question and answer model. The interviewee is normally only able to answer within a constrained manner. For this work a discussion format was chosen where both the interviewer and the interviewee worked through a series of prepared topics, with the interviewee being guided where necessary by direct questions.

In this manner the interviewee tended to be more relaxed, less intimidated by the process and free to elaborate and develop the discussion around topics of importance. At the same time, the interviewee is constrained within certain core topics giving uniformity across interviews.

The transcript of each interview was annotated and written up as soon as practicable after each interview. A presentation copy was then returned to the interviewee the next day for correction and verification. Again from both parties viewpoint this ensured nothing was recorded that was invalid or misconstrued.

3.2.2.4 Observation Work Methodology

This was carried out in two stages. Firstly general observation of staff who managed the PAS was carried out, varying from operations level in District C to non-participant observation at Information Steering Committee meetings at Districts E and G. This provided useful background material of the practicalities of maintaining such a diverse system of information provision.

The second stage was completed by all Information Managers as part of the discussion interview. Each manager was asked to complete a detailed information request using a 'think aloud' approach. This information request was a 'real life' request, obtained from a consultant at District E[2]. This

[2] This request is reproduced in Appendix Three.

provided an insight into the processes carried out in transforming a verbal, often not completely thought out request, into a computer defined query. Such information formed a key part to the principles underlying the systems development work, discussed in chapters Five and Six of this thesis.

3.2.2.5 Questionnaire Methodology

At the end of a visit to a site, a questionnaire was distributed to all staff members who came into contact with PAS or DIS[3]. Topics covered were confidence in DIS derived information and individual NHS and training backgrounds, factual or personal information that is easier and more politely obtained in this manner rather than face to face. All questionnaire responses are completely confidential and were posted direct to the University.

3.2.2.6 Subjects

Within each District the aim was to interview at least the person charged with the production of information - usually the Information Manager and the person charged with the provision of the computing environment in which the information systems were based - the Information Technology Manager. In most cases however, it has been possible to go further than this and interview at least one specialist user of this computerised information, for example the Clinical Information Assistant in District D or the Director Of Public Health in District E. From the combination of these structured discussions,

[3] This questionnaire is reproduced in Appendix Four.

observations and completed questionnaires, an overall picture of patient related information has been built up for each District.

3.3 Observations

PAS provides the cornerstone to information provision within a District. PAS holds all the basic patient and District related information, for example patient name and address, hospital details and so on. This information forms the basic information resource for the practical administration of a hospital or ward, for example providing bed state information. PAS therefore provides the basic core feeder information for any other computer systems, for example case mix management or District Information System (DIS). Alongside PAS is mounted DIS. DIS provides a complex historical database of patient activity, principally for central statistical returns and local level information reporting. DIS is therefore dependent on PAS as a feeder system, holding aggregated historical records of all patient interaction with the District. DIS provides the function of a reference library, storing vast quantities of information entered in a standard format by someone other than the final information user. In a reference library, the information although in a predefined format, for example split into chapters and indexes, is directly accessible from the open shelf.

DIS by comparison provides no direct access or methods for either browsing or searching via indexes. Vast amounts of information are held but the user has no direct means of access. Instead each District has evolved some form of Information Department which acts as an intermediary or 'gatekeeper' between the final user and the information itself. To access information, the user is required to go via this librarian.

In all of the Districts visited, requests for information are routed via this department, regardless of organisation or infrastructure employed. For example District F had a markedly different structure when compared to other Districts in the sample, with a decentralised information function, attached to individual 'units' through the District. Requests for DIS based information are still routed via a central Information Department. Similarly District B even with ward based terminals, restricts information requests of DIS to a central Information Department. Such a structure provides only one repository of information and usually one point of access.

The demand for information from DIS was established as the number of information requests made of DIS in a 'usual' working week. Note this does not mean the number of queries made of DIS, as a single request for information is often decomposed into several DIS queries. The number of requests for DIS based information recorded is presented in Table 1. It is the role of the Information Department to translate these information requests into information responses for the final user. The information manager becomes the interface itself between the final end user and the information structure.

Number Of Queries	Districts In Study						
	A	B	C	D	E	F	G
< 3			•	•			•
3-5	•				•		
> 5		•				•	

Table I Requests for DIS Based Information

The role of the information function as an intermediary can be disaggregated into two stages of interaction, firstly between the final end user and information management, and secondly between the information manager and the system itself(DIS).

3.3.1 Final User-Manager Relationship

For the user, accessibility to the information resource is provided via internal communications within the District, for example via telephone or memo contact made with the Information Department. District F had a higher interaction profile than other Districts in the sample, providing an Information Shop. Potential 'customers' for example clinicians or Unit General Managers(UGM's), could browse shopping lists of standardised reports and general abstracts of information available within the Information Department. In this environment, a customer would then be able to discuss face to face their information requirements and even browse information off the shelf directly. Information could then be ordered or borrowed directly from the shop. For all other Districts in the sample, users had no hands on direct access to the information and so may remain unaware of what information is available. By comparison to the reference library model, the problem of user awareness is identified.

Awareness of what information is available may be derived from two sources: Firstly as a byproduct of the initial information implementation process, and secondly through subsequent ad hoc means of promoting the information function carried out by the Information Department themselves. These are discussed in the following sections.

3.3.2 Installation and User Awareness

Potential for raising user awareness of the information available from DIS lay principally in the initial implementation processes carried out at District level. All Districts followed the same general implementation strategy (as defined by the Region), however some were more successful at raising the level of user awareness of what is available.

3.3.2.1 Generalised Installation Process Followed:

The implementation process followed was based on the Inter Regional Collaboration (IRC) with ICL co-operation in systems design and hardware support. The basis of this approach was to develop a PAS on a 'representative' District which would then be provided as a 'turn-key' package to other Districts who participated in the initiative. In whole or in part, over eighty ICL patient administration packages have been installed in the UK. Such an approach aimed to limit the disruptive effects of systems analysis and design to a single District, minimising the confusion and workload problems of parallel running. This was especially important as many Districts were in turmoil, only just establishing a computerised information function, for example Districts D and G. Providing a turn key approach enabled development work to be controlled from a central coordinated position, rather than over a piecemeal spread of Districts. It is unlikely that each individual District would have had enough resources or experience to embark successfully on such an extensive programme of software development.

However this approach rests on two key assumptions: all Districts have a similar internal organisation or infrastructure, and are of similar profile. In practical terms it was assumed that each District had a single District General Hospital(DGH) at which a centralised reporting function was provided. This is not the case for a number of Districts in the country, including Districts C, F and G in the sample taken for this study. These Districts have two or more acute sites, complicating further information provision. Similarly the District profile is intended to be similar for all Districts. This is also not the case, when comparing a London teaching hospital based District to a more 'rural' based District, for example District G where markedly different information infrastructures exist. A second example is found in District B, which contains split local authority wards but which the DIS datamodel assumes as whole.

In addition, this approach locks each District into a single supplier, tieng to a single design structure, for example ICL asynchronous communications protocol when all other software installed in each District used synchronous communications. However contracting to purchase in bulk in this way offers economies of scale in terms of fee and user support for Region. This regional initiative generated significant protest at District level, especially from those Districts with established patient administration systems, for example the two London sites (Districts B and F).

All Districts had some form of PAS in operation prior to the implementation of a standardised, computer based PAS. For some (Districts A, C, G and E) this was a manual card index with associated problems of lost or misfiled cards, extensive duplication of information or poor handwriting. From the sample, Districts F and D were partway through computerisation, whilst District B (a teaching hospital site) had an extensively computerised PAS installed, operational and accessible from ward based terminals, supported by inhouse programming staff.

The implementation process reported from each District was less than smooth, with each reporting missed implementation targets and data entry backlogs. For example District D had an NCR PAS based system on which patient records were being entered at the time the ICL DIS and PAS products were to have taken over. The transfer of punched data to the ICL format was completed at Southampton DHA. Ten percent of records were lost in the conversion process. This was from an existing shortfall of approximately 40% of total. By the date of the study there were still trying to catch up, with an estimated 20% of patient records missing.

Having a previous computer information system installed did not appear to directly affect user awareness of the information resource for ad hoc management decision support. Prior to implementation, requests for information were reported as minimal in all Districts, except for the London teaching sites which generated research orientated queries. This can be attributed to the relatively low profile given to management information until the Korner initiative of the early 1980's. For example

> 'there has always been plenty of operational data about. Before anyone could have requested information simply by 1/ asking for it, 2/ pushing for it, and 3/ presenting a case of why it would be useful' (Information and Computing Manager District D)

However through the process of implementation there was opportunity to build on the high profile established by the Korner initiative. This could have been carried out by promoting the information function, involving as many people as possible through the District in its implementation. This was successfully carried out at District F, which mounted a complete public relations exercise, promoting the new systems and information available to existing staff. This raised general computer awareness and appreciation, resulting in a surprisingly smooth transition to the ICL based PAS and DIS.

The other London District, District B, has maintained an equally high computer awareness and appreciation through a combination of clinician training and involvement in computer application development, for example by having a Computer Advisory Committee chaired by a Professor of Medicine. This District had a more established PAS, installed in the late 1970's, called a 'telefile' system. Ward based terminal access to patient information was provided, supported by inhouse programming staff who could tailor the programs close to District or even individual clinician specification. These specifications were arrived at by a high level of clinician involvement in the systems analysis and design of new applications. This interest is bolstered by extensive inhouse training: Each new Doctor or nurse receives a two day computer appreciation course in hospital information systems with training supplied inhouse by five full time trainers in dedicated premises. District B had rejected the IRC/ICL initiative and at the time of the study, several alternative suppliers for the replacement for this telefile system were being considered. Here computer awareness was maintained principally through extensive clinician involvement in inhouse application development, supported by compulsory initial computer training.

In both of these Districts, increased user participation in subsequent computing projects plus a high level of interest in information within the District is maintained.

The Region identified training as a key component of the implementation process. Due to resource constraints, Region advocated 'cascade' training. This approach removed staff from the work setting and intensively trained them offsite. On completion, these staff were returned to the workplace to train up co-workers and new workers as and when they arrived. Such an approach however assumes that the trained worker stays with the organisation, that is staff turnover is low. As an example, in recent years, District D has experienced 50-60% staff turnover per annum in the medical records function.

All Districts identified other practical problems with this approach. On return after training, the trainee will be under pressure to resume job function, often having to catch up the backlog accumulated whilst away. In addition there is often not sufficient time to pass on acquired knowledge to new or existing members of staff. Often a trainee will only absorb that training which directly related to his/her own job function or personal interest, usually picking up bad habits along the way. Trainees found difficulty in recalling information, but especially if information not directly related to their own need. Ultimately the trainee is unlikely to have any experience or qualifications in training others. Personal and political problems are therefore likely to compound this filtration of knowledge, for example a younger woman may possibly alienate an older woman during training, an example which occurred in District D. In summary, all Districts reported this approach as unsuccessful.

This study identified that those Districts which had gone further than the standardised Regional implementation and involved information users directly in implementation and strongly publicised the changes in information provision, had a higher demand for information and hence utilisation of the DIS resource. Essentially clinicians were more aware of what information was available in these Districts, and thus had more opportunity to utilise what was available.

3.3.3 Post Implementation User Awareness

The placement of the information function within the structure of the District can affect the development and emphasis of the information function itself. For example in Districts A, B, C and E whose line management were Public or Community Medicine, the provision and use of information relating to applied health matters was disproportionately stronger, for example information was distributed via the production of health reports and information compendiums. Where the provision of information is merged with the computing/IT function (District G) or the treasury function (District D) information provision was comparatively weaker, lacking in application drive. The provision of information to manage health matters was therefore better provided where the information function is responsible to a health function, rather than to an administration function.

This placement reflected also the level of information promotion maintained within the District. Most Districts provided an initial mail shot of standardised reports available to clinicians and UGM's, which often tapered off into non-response and inactivity. District A was one of the minority which provided a regular shopping list of standardised reports which was mail shot to consultants and UGM's. A compendium of information available in the District was also generated, containing example information applications which was made available to those who expressed an interest. District E provided the same approach, but went further generating a full glossy analysis of patient health within the District for external publication.

Mechanisms to promote user awareness varied considerably between Districts, from no promotion mechanism at all (Districts C and G) to those districts that provided inhouse computer initiation for all medical staff taken on (District F) and inhouse training and advice centres(District B), health reports and information compendiums. What came through strongly from the study was those Districts which actively promoted the information function to potential users, met with a higher level of interest and subsequent higher utilisation of the information resource. Following the library model, user awareness was identified as a key factor in user accessibility, fostering user participation and utilisation of the available information resource.

3.3.4 Information Systems Manager and Department: Information Systems Relationship

This section concentrates on the relationship between the information manager and the system(DIS) itself, a relationship which is hidden from the information user. With DIS providing the information resource or library of information, the task of the information manager is to act as an intermediary, translating the end users textual request into an information report.

Four key areas of accessibility were identified, relating the Information Manager to the information resource:

- Practical accessibility problems,
- Staff training and awareness
- Data model complexity and
- Interface complexity.

3.3.4.1 Practical Accessibility Problems Identified:

Each District was provided with DIS and PAS software, mounted on the same ICL2960 hardware. This supported remote access over an asynchronous communications network for up to forty remote terminals. Due to resource constraints and potential organisational upheaval, the complete ICL Patient System was not implemented simultaneously, but rather in a staged manner. Districts were allowed to proceed at their own pace towards broadly set deadlines.

All of the components of the patient system were to be mounted upon this same processor in each District. All Districts in the sample had the Patient Master Index(PMI) and DIS modules installed and were in the process of implementing both the outpatient and waiting list modules. The exception was District F which had waiting list, PMI, and outpatients installed simultaneously to replace those functions already computerised. District B had

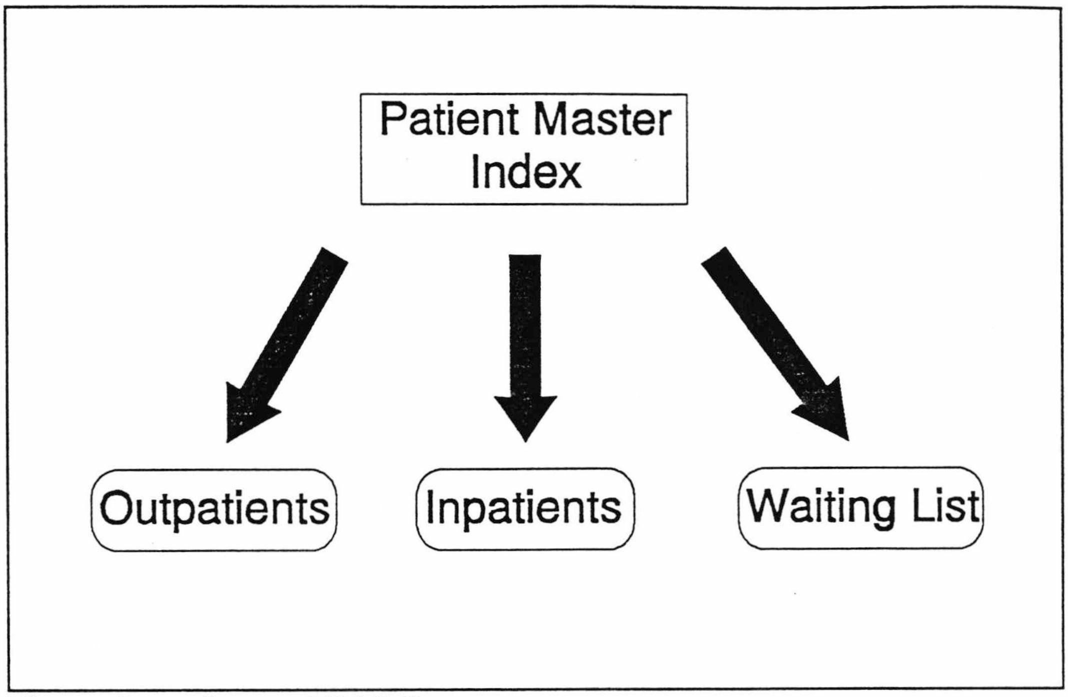


Figure 8 Internal PAS Components

its own PAS software, but had taken DIS from IRC/ICL.

The structure of the ICL/IRC patient administration application required two

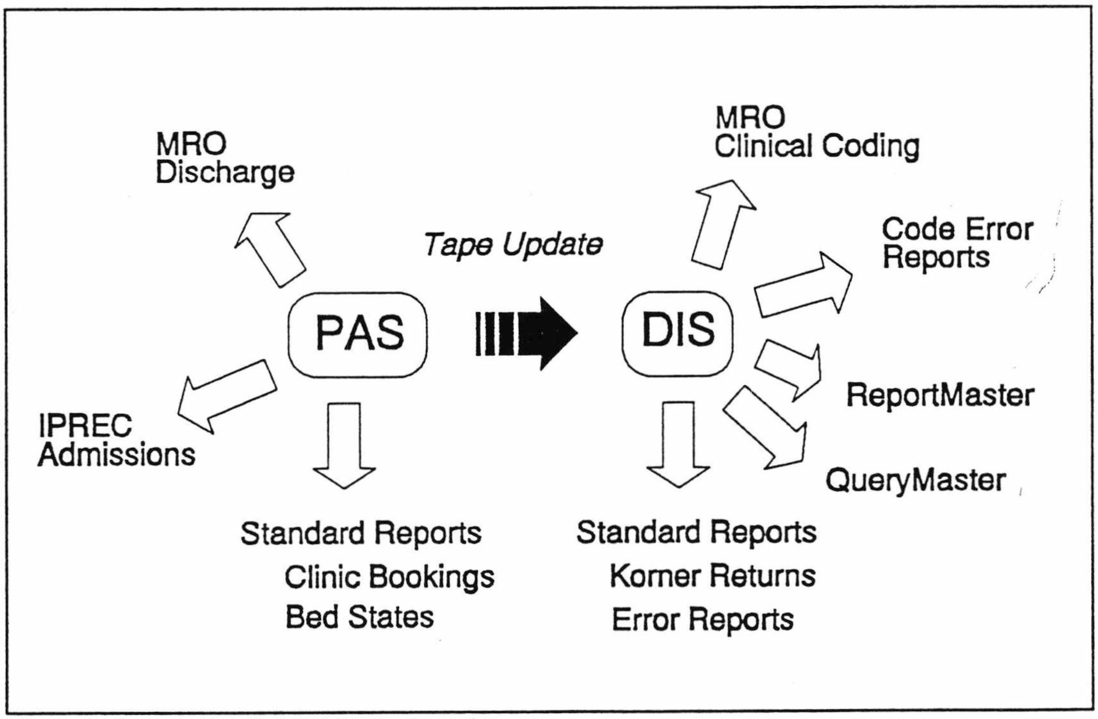


Figure 9 PAS-DIS Relationship

separate databases, PAS and DIS with DIS holding substantially the same data as PAS. PAS contained all the administrative data relating to patients, whilst DIS is the core components of this data plus clinical information, entered as clinical and diagnostic codes from International Classification of Diseases Number 9(ICD9)[99].

Several operational problems arise from this structure, for example modifying records in PAS had to be reflected in DIS and vice versa. However the key problem with this structure was that medical records and admissions staff use PAS whilst simultaneously Information Management use DIS. On patient arrival, PAS generates an IPREC[4] form containing details of admission. On discharge, the diagnosis and operation codes derived from the clinicians notes are written on this same IPREC form, the patient is discharged from PAS and clinical codes entered directly onto DIS. This process is carried out weekdays, 9 - 5pm by medical records clerks at hospital site level. Admissions staff require access to PAS during this same period. In the future when patients admitted via A&E are entered directly onto the system, admissions will be required 24 hours a day. During this period information management is likely to require access to run queries and generate information reports. Thereby three independent groups require access to the same patient information.

The IDMS database application was equipped with read and lock integrity protection during query generation, prevented simultaneous access to entities

[4] IPREC: An In-Patient Record form, an example is provided in Appendix Five.

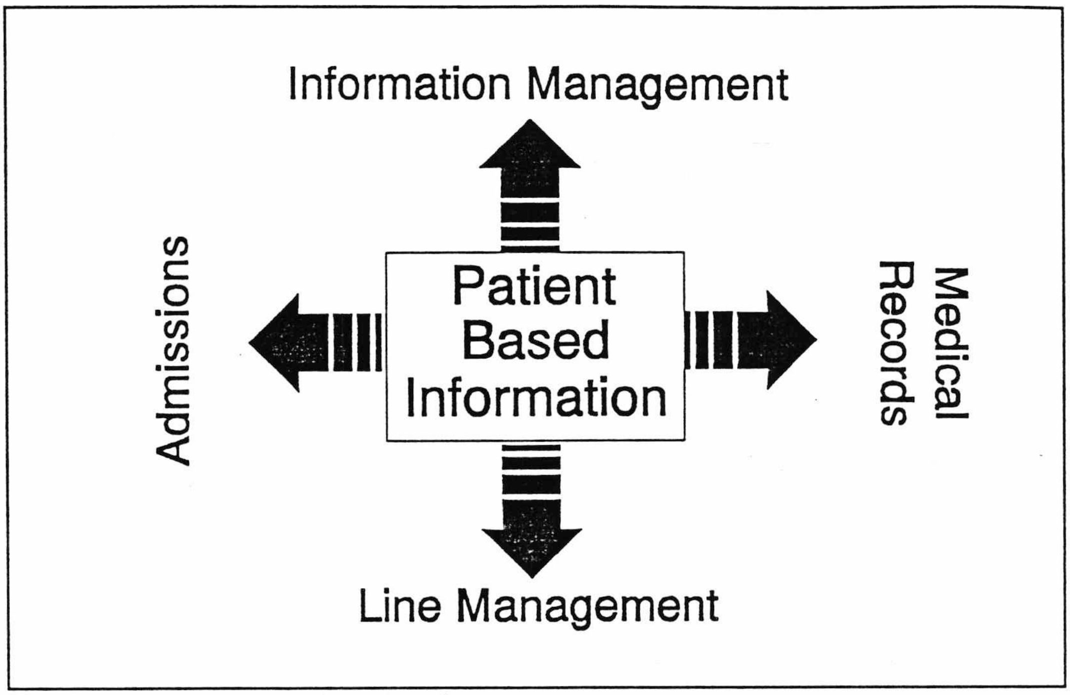


Figure 10 Users of PAS Information

required in a query by another user. During the processing of a query, this had two seriously damaging side effects. Firstly this prevented access by Medical Records staff, entering diagnostic codes onto DIS. Secondly such action sufficiently slowed the machine to effectively prevent admissions generating IPREC forms, that is admitting patients to the hospital. It was envisaged that these problems would become more acute as more outpatient clinics were added. District G was the most advanced in outpatient module implementation in the Region, having half of the 200 clinics installed. They were already experiencing significant slow down problems. Additional operational pressures on the machine were becoming evident. For example, scheduling letter printing for output, and waiting list and clinic management routines were placing pressure on information management batched queries, as well as planned backups and maintenance.

In operational terms, all Districts could not run queries of DIS during normal working hours. Query development was therefore confined to overnight batch processing. Complicated query development quickly drew out into long processing times and five day development cycles, which was graphically described by Districts D and G:

Day 0: Submit query into overnight batch queue.

Day 1: Errors in coding syntax are identified and corrected before resubmission.

Day 2: 'There is no guarantee that the machine will work in the way you expect it to, so usual to get a different answer to what thought was asking for' (Information and Computing manager District D)

Day 3: The final user is often not completely sure what is asking for. Usually a vague request was specified. The information now presented to him/her is likely therefore to be not exactly what perceived as requested. The query is resubmitted.

Day 4: First valid answer available from the machine.

Day 5: Resubmit in an alternative format to verify information generated. Check also with common sense and other informed people.

Several Districts were in the process of developing their own solutions to tackle this problem. These applications were fed from standard DIS overnight batched reports. District D rekeyed output (using a YTS trainee) into Lotus123, whilst District E was developing a complete standalone Ingres database. District F used a training database (a subset of the main DIS database but mounted on the same processor) to develop correct query syntax before submitting to the main database in the overnight batch queue. This reduced the five day development cycle without interrupting medical coding, but still slowed admissions work, that is IPREC generation.

Such operational problems provided great disincentive to use, fuelling a feeling of overall lack of control over the 'phantom'(District B). This was reinforced further by the information function having responsibility for information output but having no control over either information input or intermediary processing operations. Input to the information function, that is medical record coding, is accountable to line management based at the hospital site level. Similarly processing this information cannot be tailored, being carried out within a turn key application. For example, Regional implementation of PAS was described by District D as

> 'a dog to bark at the man if he tries to touch the machine and a man to feed the dog'

This feeling manifested itself in constant checking and rechecking of information output from the system with other colleagues, the requester of the information and in extreme cases, other Districts. Several Districts were attempting to bring medical coding into their line control, for example Districts D and G. The situation was most acute however, where an Information Manager had a non Health Service background, for example District D. In this situation such a manager had no effective 'common sense' test over the information generated.

In summary, DIS information suffered from a substantial lack of credibility, culminating in an overall sense of a lack of control - 'an application running out of control' was cited by several District Information Technology Managers. This affected utilisation in two opposing ways. Firstly more requests for information were made to substantiate already derived information and secondly application utilisation fell as Information Managers 'actively discouraged' use in favour of paper based standardised information.

3.3.4.2 Staff Training and Awareness

For new staff entering the information function, the main route to an awareness of what the information systems(DIS) can provide is through training, either on formal courses or more realistically 'over the shoulder'. The formal training courses provided to the sample Districts concentrated on operational query development (Figure 5). Little contextual, information use or general computing skills training was provided (Figure 6). The training concentrated on query generation, tailored on the assumption that participants had an extensive knowledge of information provision in the Health Services setting. Based upon the questionnaire returns, the training was generally considered satisfactory by all Districts for query generation, but completely inadequate in the broader perspective of what to do with the information system as a whole.

The main observed path to training staff was over the shoulder, due to time pressures and need to get the job done. This was compounded by an overall

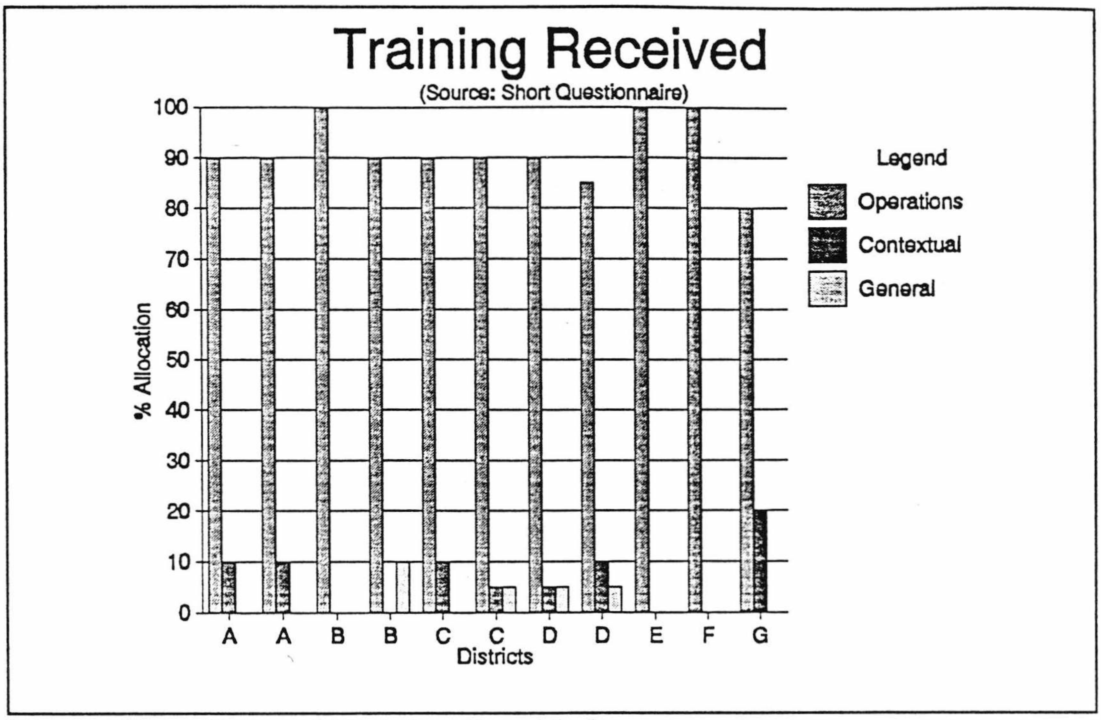


Figure 11 Form of Training Provided

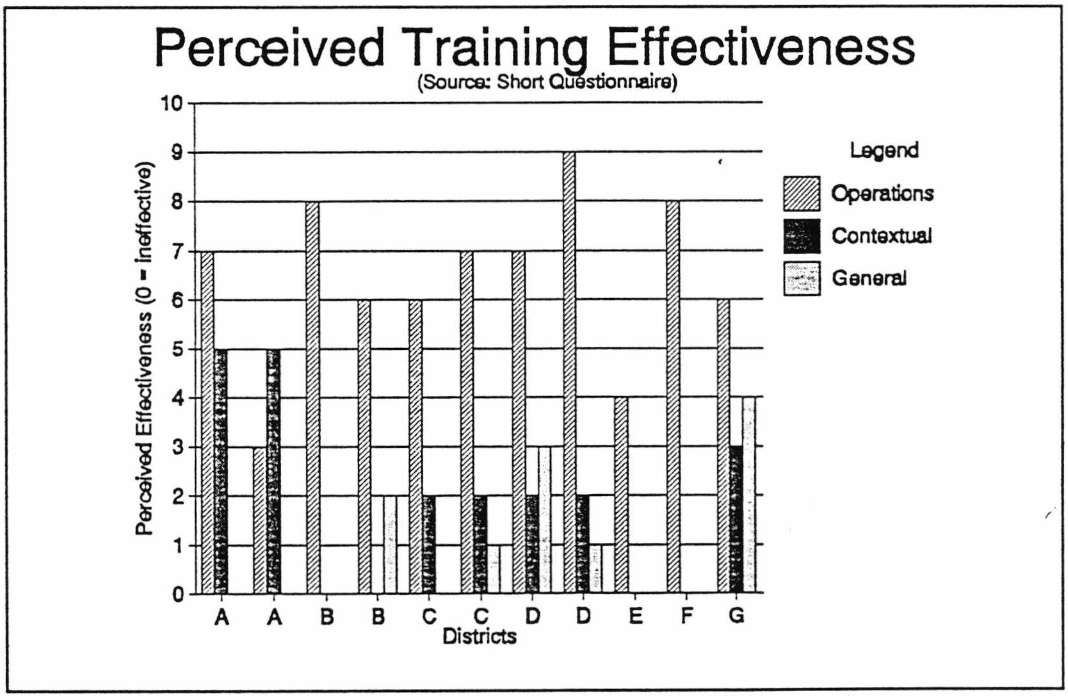


Figure 12 Perceived Effectiveness of Training Received

lack of training facilities through all but the London Districts (Districts B and F), which had their own dedicated training room plus training staff. A more typical picture of the situation outside London was District D where, because of space constraints in Victorian buildings, the training room was used as a general computing office. Overall little contextual training was provided for information system users. A key problem, considering part of the brief for the information function is to generate interest in information, acting as a focal point for use of the given resource.

3.3.4.3 Data Model Complexity

Heath care is a complicated process of provision, a comprehensive model of health care information is therefore necessarily complex. The data model which forms the basis of DIS is provided as standard throughout the Region[5]. This was again developed through the IRC/ICL collaboration upon a single representative District - West Midlands DHA. This model attempts to build a historical perspective of patient activity both to enable various statutory returns to be produced by the District (for example Korner reports) and to enable local management and researchers to obtain information to support local decision making. The basis of this model is to disaggregate patient interaction into District Spells, Ward Stays and Consultant Episodes. From these individual interactions a detailed picture of patient activity is constructed. The size, structure and detail of this model is therefore substantial. In particular the complexity of the data structures leaves the model widely regarded as unmanageable, especially as included in the structure are elements relating only to Regional aggregation. Every District identified this as the most fundamental problem in query development and

[5] This model is reproduced in Appendix Six

information systems utilisation. For example:

> 'There is a lot of interlocking, so never really sure route a request is taking. A flat file would have been more appropriate...completed a two day training course but still horrified by the overall user diagram' (District D)
>
> 'Very complicated datamodel, in fact unnecessarily so. I doubt whether anyone in the country understands it!' (District F)

Each Information Manager identified the need to make use of DIS, but qualified this with requirements such as

> 'to know the data' (District D)

and

> 'to know the data model structure, that is where to locate and combine entities to depict information' (District F).

District A summarised the position as a requirement to,

> 'keeping the finger on the pulse, monitoring the current state of the entire application'.

In response to these requirements, Information Managers were trying to maintain some confidence in the information produced by keeping to known, well trodden paths through the structure.

> Need a person who has experience of "hospital life" rather than a pure computing background. Only way to handle the complexity of the database is to experiment. In formulating an actual query identify where the fields to be retrieved are and chart a course around the structure on that basis. Try to identify tried and trusted sources and develop from these outwards around the diagram' (District D Clinical Information Officer)

Where possible, for example with sufficient resources and senior management enthusiasm, alternative approaches to this issue were being developed. For example District E was building its own data model in Ingres, in an attempt to provide an immediate source of operational information.

The size, structure and detail of the data model employed provided a significant level of structural complexity for users in all Districts from the sample. In order to compose a query correctly, the user is required to be aware of the entire data structure, the current location within it and the various paths through the structure to link up into a satisfactory query. This was identified as a fundamental accessibility problem which directly accounted for a reduced level of utilisation.

3.3.4.4 Interface Complexity

The aim of Region was to provide a standardised PAS implementation throughout all constituent Districts. Data sets were then to be uploaded onto the Regional DIS, providing an aggregate picture for centralised statistical and planing purposes. Standard ICL software was provided for query development: QueryMaster, and for report generation and presentation: ReportMaster. ReportMaster was not in use in any of the Districts in the sample. ReportMaster was uniformly considered 'too complicated to justify the learning investment' with all Districts pursuing PAS to PC file transfer to enable

> 'analysis with sensible software such as SPSS' (Information Assistant District F).

In the meantime, users often received raw QueryMaster output or for larger reports, QueryMaster output was rekeyed into a Lotus spreadsheet to provide basic graphical presentation.

QueryView was a command line query generator providing a standardised interface to all users regardless of background, experience in computing or request to be made. Similar to ReportMaster, it was viewed as unnecessarily complicated to use, for example changing the date on a report required day, month, and year to be entered backwards. QueryMaster, unlike ReportMaster however could not be avoided if use was to be made of DIS.

No help system was provided beyond the 'difficult to follow' manuals supplied. Primitive editing facilities were provided, for example the main mechanism for deleting characters was to space bar over text. However the remaining text was not reformatted on the screen, resulting in a patchwork of text and space. No screen scrolling for QueryMaster output was available, with reports greater than a single screen length wrapping round over itself. Output over a screen length was diverted to paper output.

The user was required to enter abstract computer defined field names explicitly, with accurate typing, for example K-DIST-SPELL-KEY. All user input and systems output was in upper case, leaving the screen difficult to read but also impossible to distinguish commands from textual output. No facilities for chunking or grouping information were provided, for example no provision was made to successfully refine a query upon a dedicated block of output. Instead the user had to refine the query request as required and resubmit to the batch queue again.

This combination of problems provided a formidable degree of complexity to be overcome before a user could derive information from the system. For example:

> 'user has to be a computing expert to get anything out of the system....not possible to do anything quick and nasty' (Information and Computing Manger District G)

All Districts reported this as the key reason for reduced or complete non use of DIS. District B used Wordstar (IBM PC wordprocessor) as a development environment completely offline, providing superior interface features, for example full cursor control and spell checking field identifiers. District G actively persuaded clinicians away from ad hoc DIS querying, towards standard reports and paper based information systems, avoiding DIS 'wherever possible'. However acknowledged as taking longer to answer a request for information, a sense of control over the process involved was felt to be maintained.

In District D this problem was mitigated by the Information Manager having a background in computer programming, similarly at District B computer science graduates were employed to generate queries. At District F, standard queries could be submitted to the batch queue, but the development of new QueryMaster queries was routed up through the unit structure to the Information Department. In all other Districts where Information Managers had come principally from statistics or information provision within the Service, Information Managers were either struggling or were diverting information generation to either the standard reports or paper based information.

For the Information Manager, the combined complexity of the user interface and data model structures, fuelled the already poor credibility of DIS information arising from the operational problems discussed above. For the final user, utilisation of the information resource was more a question of how successfully the information function was sold, generating user awareness and enthusiasm in what was available.

3.4 Summary

The utilisation of an information system, the amount of use an individual, group or organisation makes of an information system is a key variable in MIS research. Discussed in Chapter One, this has often been seen as the key to information system success. With the continuing investment in development and implementation of such systems the question of system utilisation takes on a practical significance. This is especially true of the NHS whose cost containment pressure can only increase over the coming years. This work has focused on accessibility to the information resource as a key variable in determining the utilisation of an information system.

Trice and Treacy provide a literature review in which utilisation is recognised as an intervening variable between user performance and the information technology. This work has suggested a second intervening variable, accessability to the information resource. It has been shown that accessibility to an information resource arises in two stages:

- between the final information user and the Information Manager and
- between the Information Manager and the information resource itself.

From the viewpoint of the final information user, accessibility to the information resource is provided on the basis of plain interest with no specialist computing skill. Given this level of available contactability, resource utilisation should have been uniformly high. This clearly was not the case. An important component of accessibility was identified - user awareness, and particularly the role of the information function in fostering this awareness.

Given the comparison of an information resource(DIS) to a reference library model, an information user may remain unaware of contained information. Those Districts which have provided user awareness mechanisms, for example computer appreciation training for all potential user staff, information shopping lists or information shops, have had a much higher utilisation of the information resource. The role of the information manager can therefore be analyised paralleling accessibility to the information resource:

- to promote user awareness of what information is available to be used and
- provide a buffer interface between the final information user and the information resource itself, principally to develop vague information requests into syntactically correct computer queries.

The promotion of user awareness as regards the final user population has been identified as a contributory factor to the overall utilisation of the information resource. The variable of user awareness plays an important part from the Information Manager's viewpoint also. Little or no contextual training has been provided to Information Management, the training provided concentrated exclusively on the procedural elements in query generation rather than query generation and application within the organisation. In practical day to day terms, information management has been forced to rely heavily on background knowledge of the Health Services. Those managers who have not come from a HS background have encountered significant problems of information interpretation, the identification of information relevance and information validity checking.

For the Information Manager, utilisation has been identified as contingent upon accessibility, that is an aggregation of interface and data model complexity. A single rigid interface (regardless of user background, demands of the system or individual knowledge), was provided for all users. In addition, the data model provided was a past, fixed interpretation by a systems analyst (that is non user) of the information resource provided. The size, structure and detail of this datamodel provided a significant level of complexity, with the user having to be aware of the entire data structure, the current location and explicit routes through the structure. Those managers who have come from a computer science background were better equipped to handle these problems. Managers from a Health Service background reported significant difficulty, the combination of a complex data model and poor interaction tools provided a significant disincentive to use the information resource. In practical terms, the Information Manager is faced with a complex data model and complex access tools with which to provide the bridge to the user community.

The next chapter begins to tackle these compounding affects of interface and data model complexity. In particular, the provision of a graphically driven, query development environment in which the user may explicitly incorporate both personnel as well as organisational knowledge into the viewing and interrogation mechanisms of the information resource. As a basis for this development, the next chapter discusses the role of the interface and the development of a 'human factors' approach.

CHAPTER FOUR

Interface Design and DataBase Querying

Contents

4.0 Interface Design and Database Querying

This chapter explains what an interface is and its basic components. The role of the interface within an application is discussed identifying a gradual progression from a software overhead or afterthought to a prime module in software development. Different visual interface styles are presented in an attempt to address the accessibility problems identified from the management research. Computer science has developed various user models, to which these interface styles are invariably tied. An alternative user model is developed for this work, upon which interface provision in chapters Five and Six is based.

4.1 Introduction

The roots of commercial computing can be traced back to the early 1960's with the installation of large, centralised systems tailored to focus on one particular routine activity, for example payroll or accountancy transaction processing. These systems were batch fed from user departments to maximise utilisation (and hence return on investment) whilst maintaining the real 'users' at a manageable distance. The overall approach at that time was to get such systems installed and operational, the end user of today did not exist. Instead information was batched and reports requested, with the actual hands on users being DP staff.

The form of interface provided for these systems reflected this usage pattern - command prompts, cryptic system commands and obscure error messages. Little in the way of 'user' help was available with users relying often on out of date operational manuals and individual technical knowledge. Given this perspective the user interface function at that time may appear to have been bolted on as an afterthought[100].

By the 1970's, many of these departmental systems had been installed in an ad hoc fashion as a route to short run efficiency savings, often in response to crisis management. From a company wide perspective such a patchwork approach often resulted in a general, incoherent information organisation with information held many times over, in different formats or media, in different locations or not at all. The development of database technology facilitated the centralisation of these diverse information sources, (through the process of normalisation) redundant data would no longer be kept and an overall information schema maintained. At the same time progress in hardware and communication technologies (notably the falling cost) facilitated remote user access lower down the organisation on a viable scale, with the real users of the information now being provided with hands on access.

At first interface design and development did not keep pace with the changes in communication accessibility and database modelling, remaining still as an afterthought for this new breed of user. Well written application systems began to turn into implementation failures as the new user community rejected adaptation to a programmer style interface[101,102]. Software engineers had all too often overlooked the interface as being too expensive and problematic to cater for, especially as this new group of users was unlikely to have any computing background. This led to a high cognitive workload for the ultimate user of the system, trying to adapt around the rigid computing framework.

This forced a change in software development methodology, away from the hard systems thinking where systems were developed under engineering principles with little or no regard for the informational content that such systems were transmitting[103], towards a user centered design methodology or human factors/HCI approach[104]. Such a change has spawned work into interface design in many areas such as task analysis[105,106] or formal techniques[107], actor roles[108], cognitive process and user modelling[109,110,111].

4.2 HCI and Human Factors

With the increasing spread of contact between user and computer based applications, both through the business/work environment but also into homes through the use of personal computing, a new research ground spawned where this 'all pervasive' technology met and interacted with the human being. Terms such as Human Factors(HF), Human Computer Interaction(HCI) or Human Computer Integration(HCI)[112] have been introduced in an attempt to label this boundary.

HCI is built around the central idea that the interface *is* the application, the requirement and nature of interaction drive the design, serving the user not the system[113]. If the user fails to achieve what he/she wants from an application it is not a failure of the user but a failure of system design. To achieve this aim, the design of an application cannot be based around an assessment of the functionality required of an application, for example a wordprocessor is *not* a collection of functions that manipulate text but an environment in which the user can succeed in generating reports and documents. Traditional mathematical and task analysis approaches are insufficient to meet this aim, concentrating heavily on the efficiency of implementation. The aim is to identify what an application should do, concentrating on maximising usability rather than implementation efficiency.

Through the 1970's user interface provision stemmed almost exclusively from the computer systems engineering(CSE) domain, often falling to secondary importance behind actual coding of the application task itself. Today it is recognised that human factors/HCI is of key importance, being a combination of several disciplines with computer systems engineering, ergonomics and psychology being the key players. The discipline however extends as far as philosophy and sociology in the work of Bjφrn Anderson[114], Frank Land[115] and Mike Cooley[4].

To illustrate the depth of the definition, Norman provides a detailed analogy with architecture. Architecture he states has both functionality within the

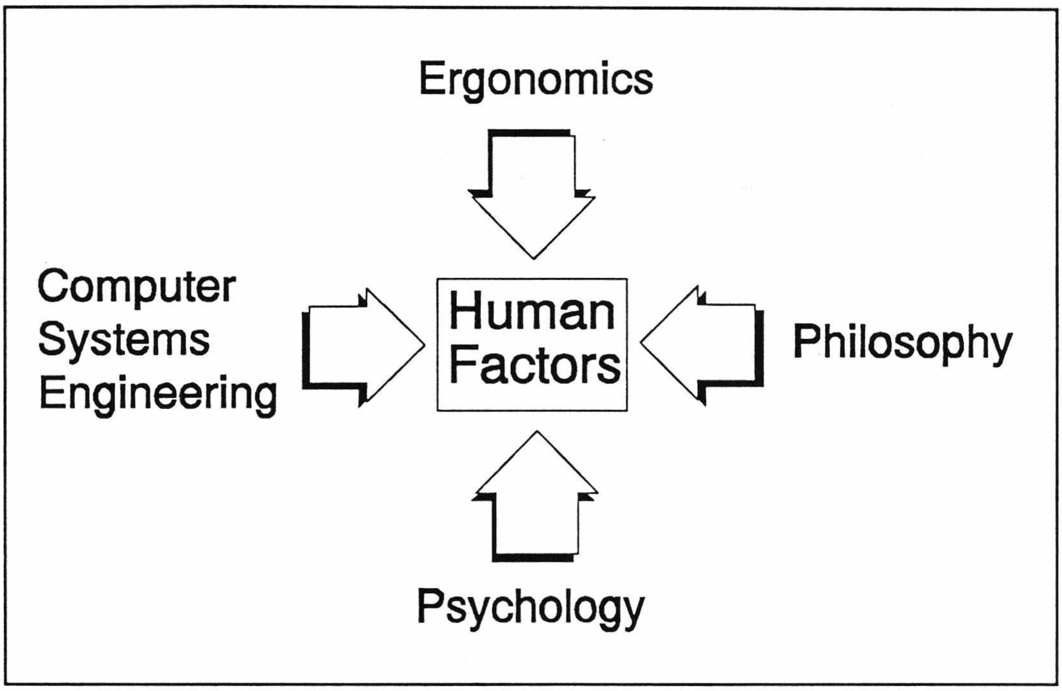


Figure 13 The Origins Of Human Factors

initial design and whilst the building is in use. The building provides presentation facade, flexibility for different owners both current and future past the initial occupier, spanning ultimately into philosophy and politics of design. In the same manner HCI has functionality both in the initial design and whilst in use, both for the initial user but also through to unforseen users and demands of the future. Architecture and HCI provision provide beauty but also shape the actions and perception of applications carried on within, providing an overall conceptual environment in which the user can work and express him/herself[116].

Research in HCI has grown rapidly, currently lying in a 'scrappy and disintegrated way'[117]. To summarise the position a pragmatic approach will be taken, splitting HCI into two interleaved aspects. Firstly the process, or dialogue undertaken by the user and application to achieve a task, and

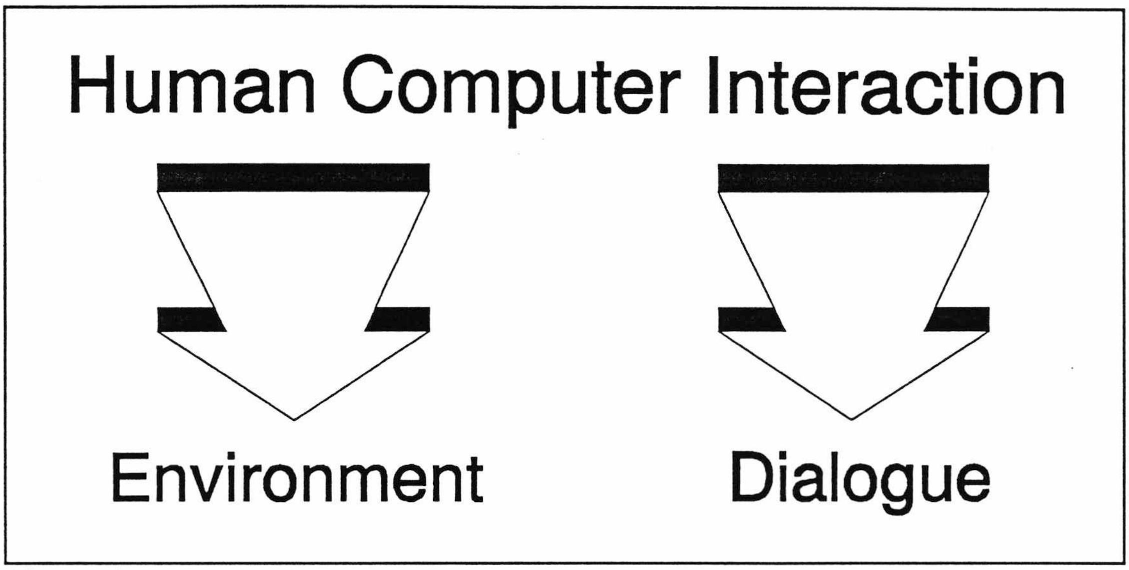


Figure 14 HCI Field

secondly the environment provided in which the dialogue takes place.

The first aspect concentrates on the user within the process, drawing work predominantly from psychology. Psychology provides many theories dealing with low level issues such as analysis of user typing error, interface learning[118], search pattern generation and font legibility. The higher level issues of user cognition present more of a demand to psychology where two main models can be identified. Firstly the Card, Moran and Newell GOMS[119] model that identified the user as a psychological entity[120] passing through a staged interaction process. This work although paralleling work in economic problem solving, for example the rational comprehensive model, traces HCI through the formulation and execution of opportunities to satisfy a user held goal. Such a hierarchial approach has been further refined and extended by Carey, Stammes and Astley[121] into the object oriented domain and by Olson and Olson into the task analysis domain[122].

Of more direct application here is the issue of directness identified by Norman, that is a feeling of natural communication between user and application. Two components of directness are identified, the gulf of execution and the gulf of evaluation.

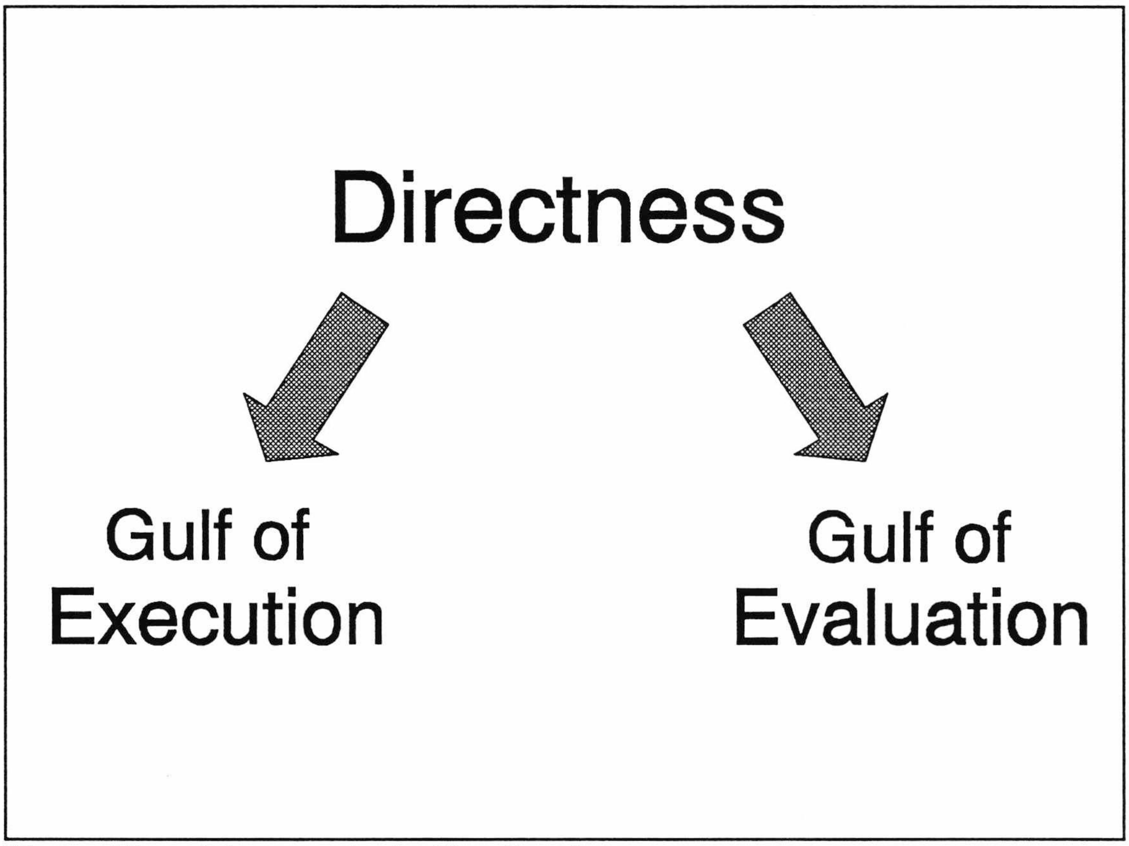


Figure 15 Directness in the User Interface

Firstly execution of control over the objects presented within the interface environment, this is best exemplified by the direct manipulation style of user interface with the user having direct control over objects within the interface itself, a feeling of 'firstpersonness'. To minimise the gulf of execution the aim of the interface is to remain transparent, providing an environment in which it appears that the user is manipulating objects from the real world. From the study in the previous chapter, such a gulf was detected, as users manipulated a poor command line interface. As a result, users tended to concentrate on manipulating the interface itself rather than the information requirement. An approach such as the NPL Electronic Paper Project[123] tackles this problem directly, aiming to shift this onus of understanding away from the user to the computer.

The second component of directness is the gulf of evaluation. The user and the system have different goals in form and context, differing in at least system knowledge (as it is assumed the user is not the systems designer). This gulf between user and system goals can be tackled either by re-expressing the structure of the interface environment towards user defined goals or moving the goals of the user closer to those of the system, for example through the use of training. From the study, this gulf was identified during query generation where the user had evaluation goals different to those of the systems analyst, who defined the structure in which the query was being expressed.

Beyond these theories the area of HCI presents a challenge to psychology, for example in the development of group decision support systems[124]. In principle the key foundations have been laid by the work of Card et al and Norman. In particular HCI should aim to implement a 'world of action rather than a language of description', releasing the user to concentrate more on the task to be solved rather than the tools (the interface to HCI) of the job.

The second aspect of HCI concentrates on the environment in which user dialogue takes place. Theory from the computing and ergonomics disciplines predominate with the definition of environment including both software and hardware. Ergonomics gives the study of person/environment concentrating on the physical side of interaction, for example reaching as far as seat construction, lighting and sound conditions, whether a mouse is better than a touch screen[125] or whether it is easier to read from a screen rather than from paper[126,127]. The software interface provides the environment in which person-machine dialogue takes place. This dialogue can be viewed as the object of HCI provision, coupled to the user via attributes or interface characteristics.

4.3 The Importance and Characteristic of Computer Interaction

The combination of interface style and environment provides the most critical part of any software application. From the users perspective it *is* the application as it is all the user actually sees, the rest is meaningless[128]. As the expert systems field has shown, an excellent system on its own does not guarantee application use, for example burying the answer behind a series/tunnel of menus is not the answer to application design[102,129,130].

From the users viewpoint it is possible to distinguish three possible attribute or characteristic groupings:

- Natural language
- Linear
- Non-linear

4.3.1 Natural Language

Natural language has been proposed as the ultimate answer to man machine communications:

> "The only way to entice a user (casual) to interact with a computerised database is to permit him free use of his native tongue" Codd[131]

The premise of this style is that the user will be free to state his requirements of an application in a familiar free form, his native language[132]. In this way the user is freed from the burden of syntax, acronyms and obscure prompts/messages and can specify in his own understanding what is required. The standard format developed would be for the user and the interface to enter a dialogue, with the interface eventually repeating back what is understood to be required. This format is useful for novice users with perhaps a vague notion of what they require, but may prove inefficient and frustrating for competent users.

The main stumbling block with this style of interface is the medium employed, the English language. Language is too verbose, laced with double meaning and ambiguity. Language is good for conversation but is inefficient for instruction[133]. Recent developments suggest a constrained form of natural language may be a practical goal for HCI design[134].

4.3.2 Linear Presentation

Linear presentation usually takes the form of a command line prompt, for example C:> in MSDOS/PCDOS[135] or QUERY> in ICL's database inquiry package QueryMaster[136]. From this prompt the user has to formulate the query/request to be applied. This is to be done using acronyms and strings of symbols that may be difficult to comprehend or may appear meaningless. By the very nature of the presentation provided (the hard systems methodology) there is very little guidance available to the user. The best that is available are help systems built around a calling function, for example on screen manual pages or keyboard help keys that can rarely provide context specific guidance.

This representation does not tie directly into the users conceptual understanding of what is happening, using only what could be described as half duplex user-application communication, that is only one line of the whole available screen space is used (as opposed to full duplex communication which would be using all the available screen)[137]. Users very quickly find difficulty in the use and understanding of such an interface, especially when job function pressure or the complexity of the application increases. For example when the query requested is abstract or vague, the user is required to be conscious of the database structure and to be able to specify explicitly in detail each component of the query before any part is processed.

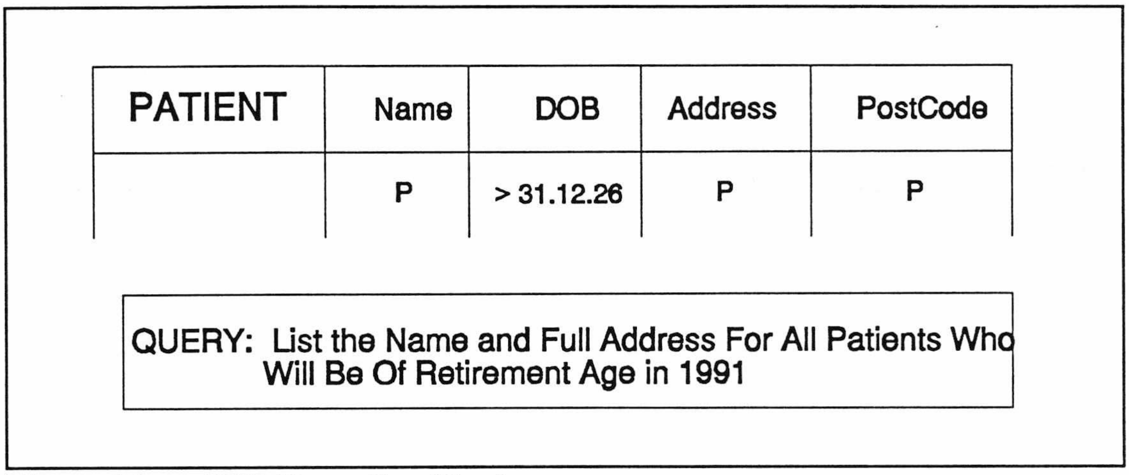
One approach to making this basic form of interface environment more manageable by the user, has been to aggregate the prompt based environment using non-linear forms and menu structures, guiding the user through into predefined frames of activity. This approach is adequate for the processing of routine inquiries/reports that can be defined beforehand for the user. Following the user definition given by Carey[138] this approach would satisfy the demands of routine processing or the mandatory staff function. For the more ill defined and unpredictable demands made by management/discretionary staff this interface becomes inadequate, leaving this group of users to specify instructions in raw programming code.

4.3.3 Non-Linear

The non-linear presentation aims to overcome the difficulties of the above linear presentation. The non linear interface style can be broken into two categories: two dimensional presentation or a complete WIMP (Windowing, Icon, Menu, Pointing-device) environment.

The two dimensional approach makes better use of the available screen space in providing the user with graphical forms for interfacing to an application, that is full duplex communications using the full communications bandwidth. For example, in interfacing to a database environment, QBE provides the user with graphical building forms in which operators, for example P for Print, example and constant elements may be placed to specify complex retrieval queries.

The form style of interface environment provides a template to guide user action. Headings provide visual prompts and the visiting order for form filling is determined by the user, allowing the decoupling of thought



PATIENT	Name	DOB	Address	PostCode
	P	> 31.12.26	P	P

QUERY: List the Name and Full Address For All Patients Who Will Be Of Retirement Age in 1991

Figure 16 QBE Query Specification

processes and query development into any order whilst keeping syntax and mathematical expression to a minimum[139]. Using this style of interface for end user database query, the issue of database structural complexity remains unaddressed - the user has to be explicitly aware of an attribute/entity location within the application structure. For example in applying selection conditions, the user is also required to be aware of the format of the data structure to complete retrieval successfully. As the complexity of the application increases, so the number and intricacy of forms required correspondingly rises incurring user mental load through inter form transfer.

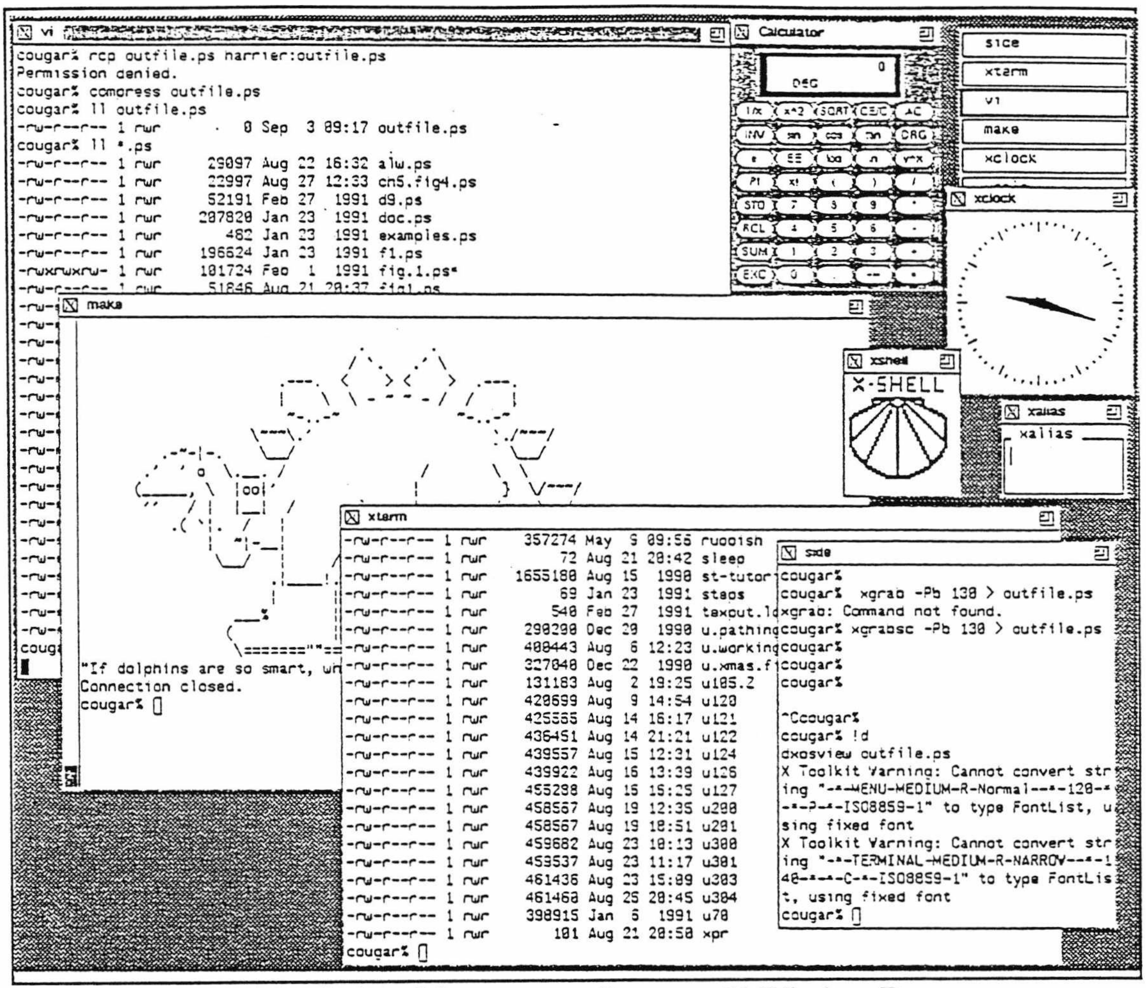
The second approach provides for complete graphical images to be displayed to the user, for example icons[140,141] or spatial maps showing the structure or format of an application. When combined with direct manipulation techniques this approach may give the user direct visual and conceptual control over an application. This environment is usually WIMP(Window, Icon, Menu, Pointing device) orientated, releasing the user from the keyboard except to input textual data. All operations in this environment have to be defined using a pointing device such that every object portrayed on screen is directly accessible. Here not only What You See Is What You Get (WYSIWYG) but also What You See Is What You Can Get Hold Of[142], and change.

Such an approach to interface design may prove restrictive, either if an application does not readily lend itself to graphical representation or if the user community actively resist change, clinging onto a prior familiar environment. For example, an engineering job manufacturing database may prove difficult to depict graphically due to the nature of the product processes being modelled. If the structure of the database is inherently complex, a shortcut for the interface designer may be to develop numerous menu or graphical structures that may then swamp and confuse the user during interaction. It may be that, for example within an accountancy database because of the user community and tasks required, the user group may be better served with a less elaborate format of interface, for example remaining form driven. In the same manner by constraining the user to strings of predefined menu options and two or three mouse buttons, the interface style may hinder the user, for example in specifying an ad hoc request it may be faster for an experienced user to specify a query in raw SQL code rather than manipulate a complex layered menu structure.

One advantage of a menu structure that is easily overlooked, is that the user is constrained within definable limits of choice. The users mental planning load is therefore also constrained, as the number of possible decisions is explicitly declared. With a direct manipulation form of interface an increased number of user decisions are required, the interface is more open, requiring more planning of the user. This advantage though must be tempered by a slower user choice, especially when either context searching through structures of menus or deciding between abstract menu titles[143].

The main approach to application design employing this non linear environment style has been the development of multiple window management systems. Multiple window management systems provide the user with any number of independently controlled graphical screen spaces or 'views'. Each view may then contain different aspects of the same application or independent applications(in a multi tasking environment). Many features incorporated in such systems stem from the pioneering work of the Smalltalk group from Palo Alto Research Centre. Features such as window management, use of graphic symbols (icons), direct manipulation and the object orientated style of programming are slowly becoming standardised.

The principal component of a multi window management system is a library of screen routines, for example view overlapping and control management, that are made available for reuse over many applications. Application development time is reduced through code reuse, but also as a byproduct over multiple applications, a standardised look and feel is generated. This reduces the end user learning curve when moving from application to application, for example all applications written for the Apple Macintosh computer conform



```
cougar% rcp outfile.ps harrier:outfile.ps
Permission denied.
cougar% comoress outfile.ps
cougar% 11 outfile.ps
-rw-r--r-- 1 rwr        0 Sep  3 09:17 outfile.ps
cougar% 11 *.ps
-rw-r--r-- 1 rwr    29097 Aug 22 16:32 alw.ps
-rw-r--r-- 1 rwr    22997 Aug 27 12:33 cn5.fig4.ps
-rw-r--r-- 1 rwr    52191 Feb 27  1991 d9.ps
-rw-r--r-- 1 rwr   287828 Jan 23  1991 doc.ps
-rw-r--r-- 1 rwr      482 Jan 23  1991 examples.ps
-rw-r--r-- 1 rwr   196624 Jan 23  1991 f1.ps
-rwxrwxrw- 1 rwr   181724 Feb  1  1991 fig.1.ps*
-rw-r--r-- 1 rwr    51846 Aug 21 20:37 fig1.ps
```

```
"If dolphins are so smart, wh
Connection closed.
cougar% []
```

```
-rw-r--r-- 1 rwr   357274 May  9 09:55 ruooish
-rw-r--r-- 1 rwr       72 Aug 21 20:42 sleep
-rw-r--r-- 1 rwr  1655180 Aug 15  1990 st-tutor
-rw-r--r-- 1 rwr       69 Jan 23  1991 steps
-rw-r--r-- 1 rwr      540 Feb 27  1991 texout.1d
-rw-r--r-- 1 rwr   298298 Dec 29  1990 u.pathing
-rw-r--r-- 1 rwr   408443 Aug  6 12:23 u.working
-rw-r--r-- 1 rwr   327048 Dec 22  1990 u.xmas.f
-rw-r--r-- 1 rwr   131183 Aug  2 19:25 u105.Z
-rw-r--r-- 1 rwr   420699 Aug  9 14:54 u120
-rw-r--r-- 1 rwr   425565 Aug 14 16:17 u121
-rw-r--r-- 1 rwr   436451 Aug 14 21:21 u122
-rw-r--r-- 1 rwr   439557 Aug 15 12:31 u124
-rw-r--r-- 1 rwr   439922 Aug 16 13:39 u126
-rw-r--r-- 1 rwr   455288 Aug 15 15:25 u127
-rw-r--r-- 1 rwr   458567 Aug 19 12:35 u200
-rw-r--r-- 1 rwr   458567 Aug 19 18:51 u201
-rw-r--r-- 1 rwr   459682 Aug 23 10:13 u300
-rw-r--r-- 1 rwr   459537 Aug 23 11:17 u301
-rw-r--r-- 1 rwr   461436 Aug 23 15:09 u303
-rw-r--r-- 1 rwr   461460 Aug 25 20:45 u304
-rw-r--r-- 1 rwr   398915 Jan  6  1991 u70
-rw-r--r-- 1 rwr      101 Aug 21 20:50 xpr
cougar% []
```

```
cougar%
cougar%  xgrab -Pb 130 > outfile.ps
xgrab: Command not found.
cougar% xgrabsc -Pb 130 > outfile.ps
cougar%
cougar%
cougar%

^Ccougar%
cougar% !d
dxosview outfile.ps
X Toolkit Warning: Cannot convert str
ing "-*-MENU-MEDIUM-R-Normal--*-120-*
-*-P-*-ISO8859-1" to type FontList, u
sing fixed font
X Toolkit Warning: Cannot convert str
ing "-*-TERMINAL-MEDIUM-R-NARROW--*-1
40-*-*-C-*-ISO8859-1" to type FontLis
t, using fixed font
cougar% []
```

Figure 17 Multi Window Management System - X Window Dump

to the same WIMP environmental style. Examples of 'toolbox' systems include Microsoft Windows and Mac Toolbox for personal computers, NeWS and X Windows for workstation based hardware.

The next step in interface design is the provision of User Interface Management System (UIMS)[144,145] facilities that attempt to detach completely the application from the interface, forming an intermediate module that mediates transactions from the user and the interactive application[146]. Such an approach gives possibility for several different styles of interface to a single application, as well as common interface provision over multiple

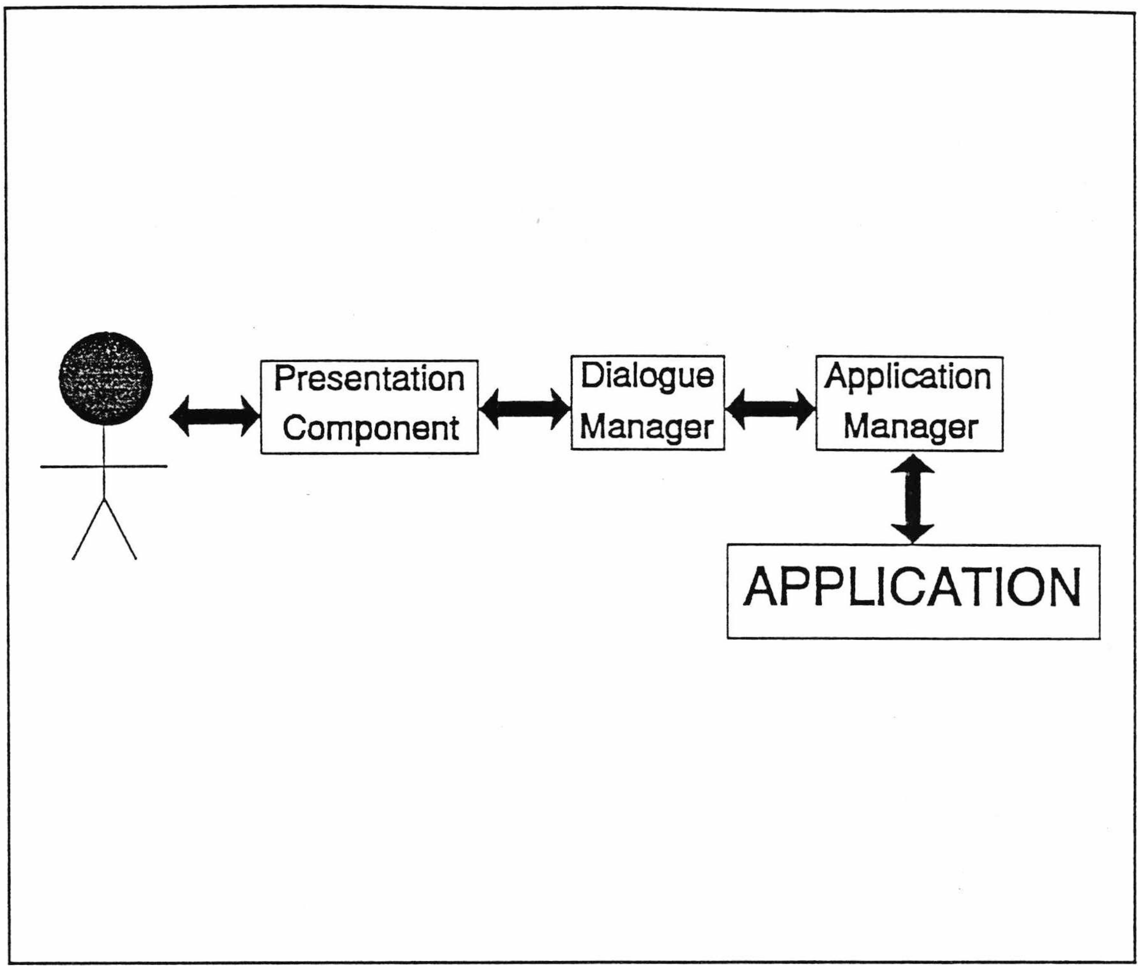


Figure 18 Typical UIMS Construction

applications. The main difference between UIMS and multiple window based systems is the provision of a dialogue definition language built around a specification model, for example state transition networks[147,148], context free grammar or event based models.

The primary advantage of such an approach is that the application is freed from low level interface operation, for example insulating the application from mouse movements providing for separate re-configuration of both the application or the interface independently. On this platform a large body of work has been aimed at the rapid prototyping area of interface design for example CHISM[149] or TIGER which provides a User Interface Management toolkit comprising 'What You See Is What You Prototype Now'[150].

The full screen communication bandwidth which these approaches exploit has provided for major advances away from the original command line 'user' interfaces of the 1960/70's[151]. One of the first implementations within a truly graphical environment was based on metaphor[152] design, mimicking on screen a real world environment or process, for example the MAC desktop style that relies on user awareness of a physical desktop.

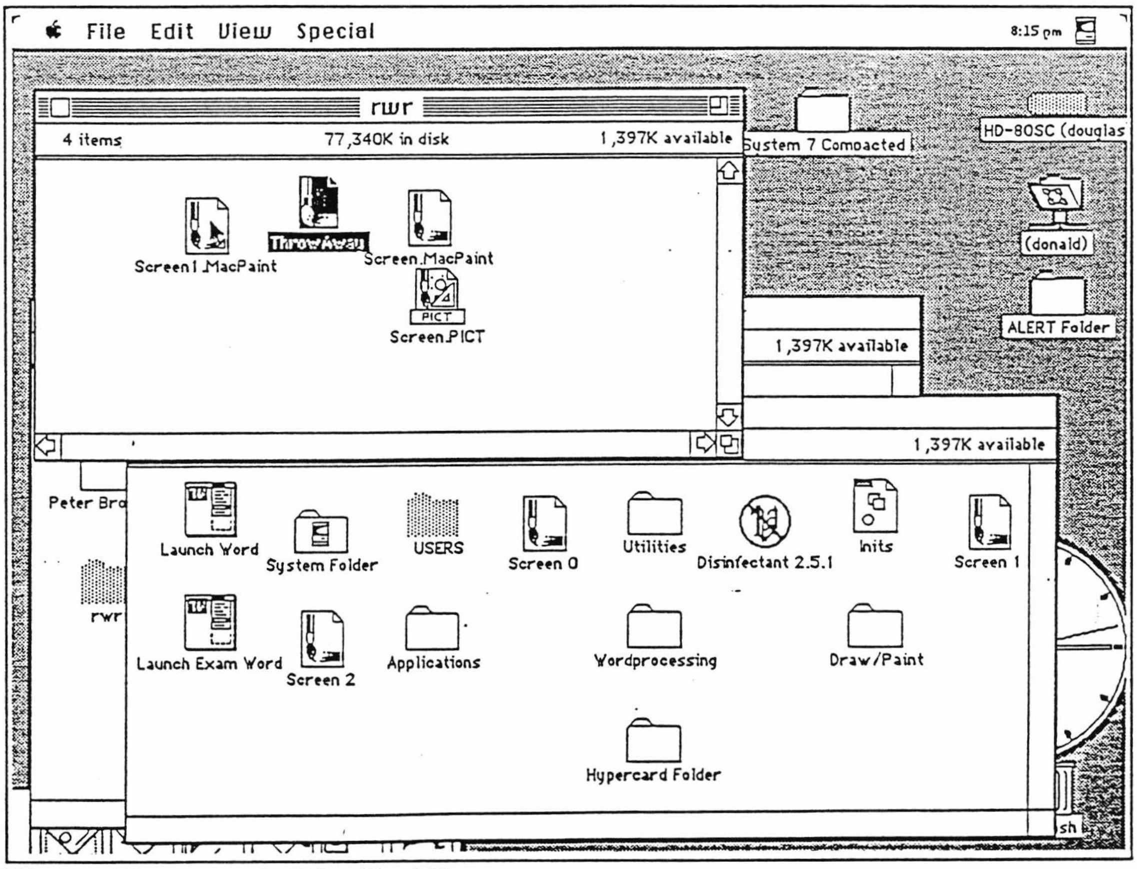
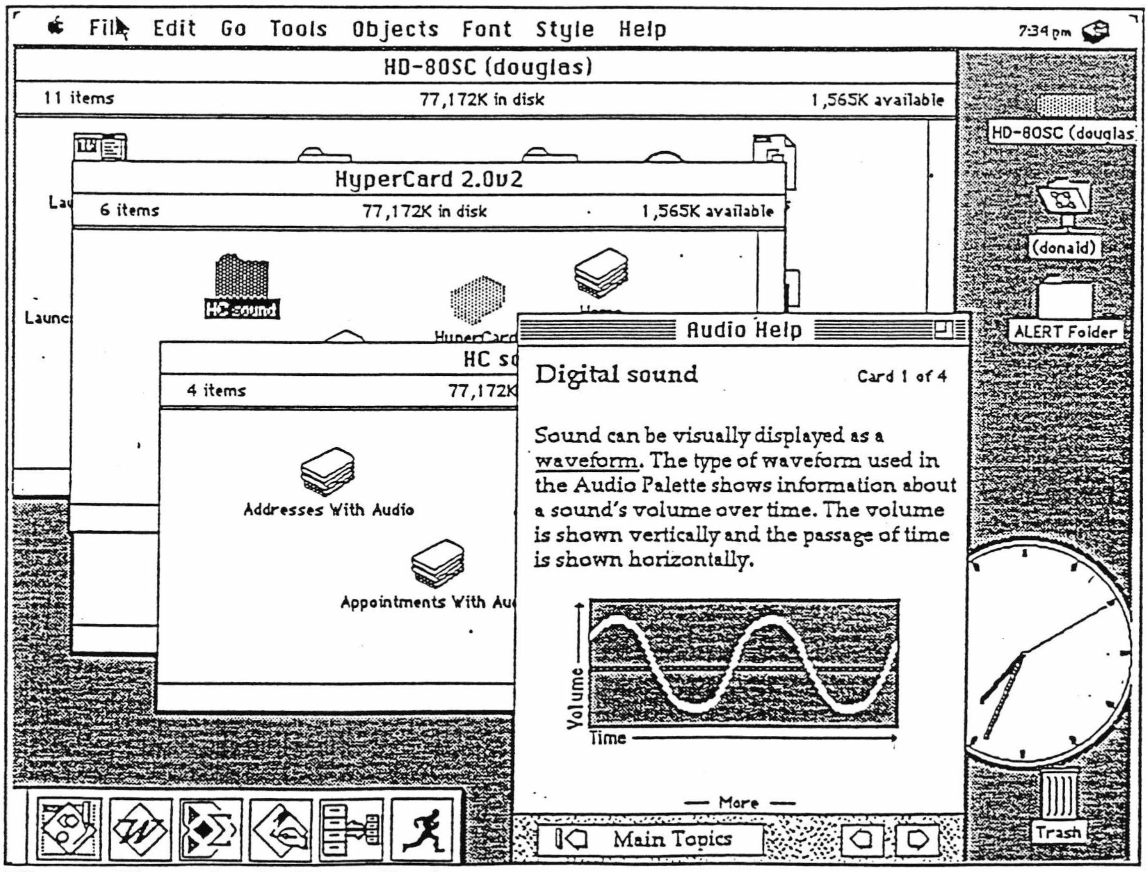


Figure 19 Apple Mac DeskTop

Good examples of this are the hypertext applications implemented with Hypercard[153] as the programming language within the standard Apple

Macintosh application environment. HyperCard provides a full screen graphics approach, based around a card stack metaphor in which the user manipulates a body of information in a self determined order. Any form of information, for example text or graphics is arranged in chunks, placed on individual cards that are then aggregated together into a card stack, via 'buttonlinks'. Buttonlinks provide a mechanism for traversing from card to card through the card stack viewing a single card at a time in any user defined order. Subsidiary links provide a mechanism giving more detail about a defined topic/graphical object, importing the contents of a target card directly into the current card.



Digital sound

Sound can be visually displayed as a waveform. The type of waveform used in the Audio Palette shows information about a sound's volume over time. The volume is shown vertically and the passage of time is shown horizontally.

Figure 20 Apple HyperCard

Using these tools the card stack metaphor provides a simple abstraction mechanism as well as a methodology for traversing the information structure.

Such a system is ideal therefore for thought construction and prototyping but relies heavily on keyword search algorithms for effective retrieval if the exact location of any piece if information is unknown[154]. In addition many applications are not suited to this construction metaphor, for example the above engineering or accounting applications.

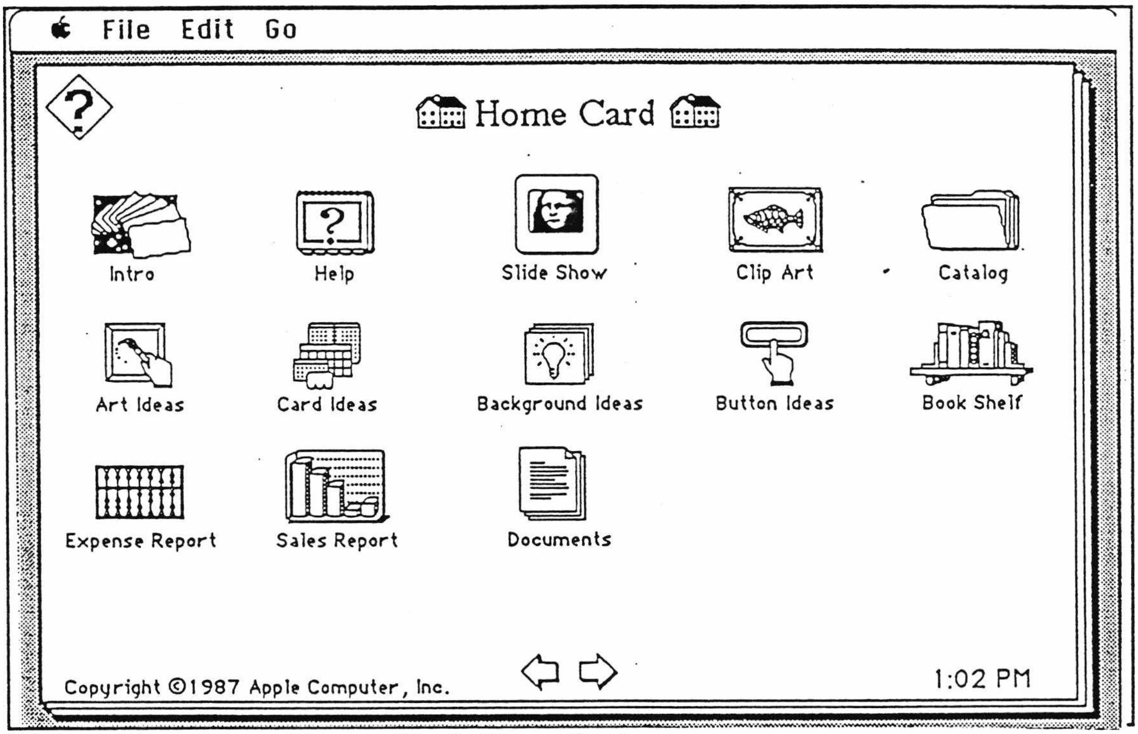


Figure 21 Front Map of Apple HyperCard

Limited routing and navigation processes are available, effectively requiring the user to move from card to card until the correct card is located. However as the number of cards and ideas expand, the required tracking of each card becomes increasingly problematic - at best the user has to rely on a spatial front diagram of card arrangement. Once more than several cards are entered into the system, the complexity of both the organisation structure and routing/navigational mechanisms, for example list of keywords, provide a disproportionate increase in user cognitive load.

A second example of a close metaphor application is the EBook[155] or electronic dynamic book[156]. Here information can be accessed through several indexes following closely the encyclopedia metaphor, for example through table of contents, subject and author indexes, whilst always being backed up by a sequential search. The issue of complexity is mitigated by these indexes, but substantial overheads are incurred in processing and storage, every item of potential interest having to be multi indexed based on some established user model.

In designing user-application communication, consideration has not only to be given to the observable surface characteristics[157] of the interface, but also to the invisible cognitive and other processes that occur when a message is interpreted by a human processor. A good visual presentation should therefore be a combination of text and pictures - a 'lexivisual' presentation. A third example use of metaphor design is provided in comic[158] construction and use. The use of comics as a communication medium provides this lexivisual presentation, combining a rich variety of what can be said coupled within a limited self contained means, that is vocabulary.

The most recent development in the metaphor/graphical interface HCI combination is Rooms from Xerox Parc[159]. A graphical presentation of an office layout is provided complete with icons depicting doorways through into other parts of the roomed structure. The user is encouraged to travel between rooms, carrying and depositing icons in his/her pockets. Tied to the object oriented paradigm, an icon can be anything from a file of text to a complete application. Splitting information and applications over the available rooms provides a chunking mechanism based on the office structure metaphor. However in travelling through the structure, mental load is incurred similar to that in using hypertext/Card, for example in remembering where icons were deposited, the current location and general navigation within the structure.

Still maintaining this metaphor based design, an early use of the Direct Manipulation environment was the development of spatial maps[160,161]. These are often suggested as an aid to database query interface design, providing the representation of data and semantic structures in two dimensional form. This allows the interface designer more flexibility in the use of clustering or neighbourhood designs as an implicit mechanism of association. Explicit arcs are available, linking associated entities whilst selective use of colour and highlighting provide mechanisms to attract attention or communicate possible action, for example shaded out menus may signify non availability whereas colour may represent possible actions or a footprint of where visited in the spatial structure[162]. However the issue of database structural complexity is still only partially addressed by this approach, providing the user with spatial maps or menu structures still requires user familiarity with the underlying data structure.

As the size and complexity of the graphical presentation increases the effective use of maps is reduced[163], for example the same problems as with Hypertext arise, for example navigation[164] and 'getting lost'[165]. Additional problems of space contention in the small screen space are especially acute. For example as the size of the depicted structure grows, it becomes increasingly likely for arcs to cross and crowding of objects eventually forcing some offscreen. The only solution to such a clustered display is to break it up over several windows - a large cognitive load for users to remember where differing objects are located. Such a solution may also lead to other problems, such as window thrashing where the user has to consciously attempt to maintain an operating window on top of the pile.

Several forms of visual interaction style have been presented in this section in an effort to address the accessibility issue. Of particular interest have been the graphical based format of user interaction, which provide the greatest scope for prototype development. One of the key tools employed in interface design has been to incorporate at least some limited form of user profile into the interface construction. The main vehicle for this has been through the use of conceptual modelling.

4.4 User Modelling and the Changing Role of the Interface

During the past two decades the role of the user interface has changed little, bridging user and application. Interest and recognition of its importance in system utilisation and implementation have, however, changed out of all recognition, moving interface provision from an afterthought to centre stage in application design and development[166].

In an effort to take some account of the user within the design process, work in this area has often involved developing precise user interface designs, targeted at specific canonical user models. However in developing a user model, individuals are all different, being a culmination of social, moral and political inputs. Every individual is therefore unique, being driven by different motivation, background and environmental considerations. The basis of development for any model is a process of abstraction away from reality to a descriptive form. This becomes problematic as writers have attempted to apply abstract labelling to describe specific user characteristics, based upon subjective group allocations. Two approaches will be presented where this has been applied: The first is based on providing an interface on the criteria of computer usage, the second is based on the task a user is required to perform using that interface.

Shneiderman[167], when addressing this problem of user definition, allocates users based on computing experience, generating such classifications as novice, experienced and intermediate. This allocation is performed on criteria such as learning time, performance, speed, error rate, subjective satisfaction and retention period of operations performed. Codd[131] defines users in the same manner, a user being either casual, non-trained intermittent or skilled infrequent. Zloof[139] identifies that these allocations are based upon fuzzy variables, but still goes on to define users in much the same terms as Codd and Shneiderman above, for example as either casual, non-programming professionals or application programmers.

Any interface built around such a canonical model would be necessarily general in nature, providing a common denominator of functionality required by all users within that particular group allocation. To gain the maximum benefit, such a descriptive user model would need to be individualised, in an attempt to capture individual users knowledge, skill and purpose in using the interface[168,169]. Two immediate problems present themselves with this approach: Firstly the problem of initial group allocation when using an interface, for example would a prospective user have to sit an interface test to determine computing competence or determine personally based on intuition? Secondly as users become more proficient, either by practice or interest, how and when would the user be switched between groups?

One solution to this is to provide an 'adaptable' interface, presenting a 'soft facade'[170] that is updated as the user progresses in experience[171,172]. This may be adaptable on a manual or automatic basis, for example based on a derived user profile[173]. Such an approach immediately leads to problems such as who should do the updating, the user him/herself or a DP professional?[174] When should the updating be carried out, for example based on a test or distilled from the user based on repeated user actions?[175] Why and how should the interface be adapted between different styles or presentation formats?

This form of updating the interface character as the user changes in experience, may lead to a problem of interface stability, especially if this updating is performed automatically[176]. Such a situation where the interface is constantly changing in nature in response to changing user views and requirements or where the interface adapts automatically but not so far as the user would like, may well generate more user dissatisfaction and frustration than if a fixed format interface had been originally applied.

Carey suggests that an interface designer should be looking further than simply listing user characteristics, building a system that matches these as closely as possible. He suggests analyzing what use the applications will be put to, rather than concentrating on user characteristics. Labels are provided, such as 'mandatory' users based on a high level of requirements predictability, for example clerical staff or 'discretionary' users based on a correspondingly low level of predictability, for example management. Panko and Sprague[177] develop this same concept using abstract name types: type1 and type2 users, to detract away from ambiguity and confusion in the descriptive titles used. The style of interface presentation would then be based primarily on this criterion.

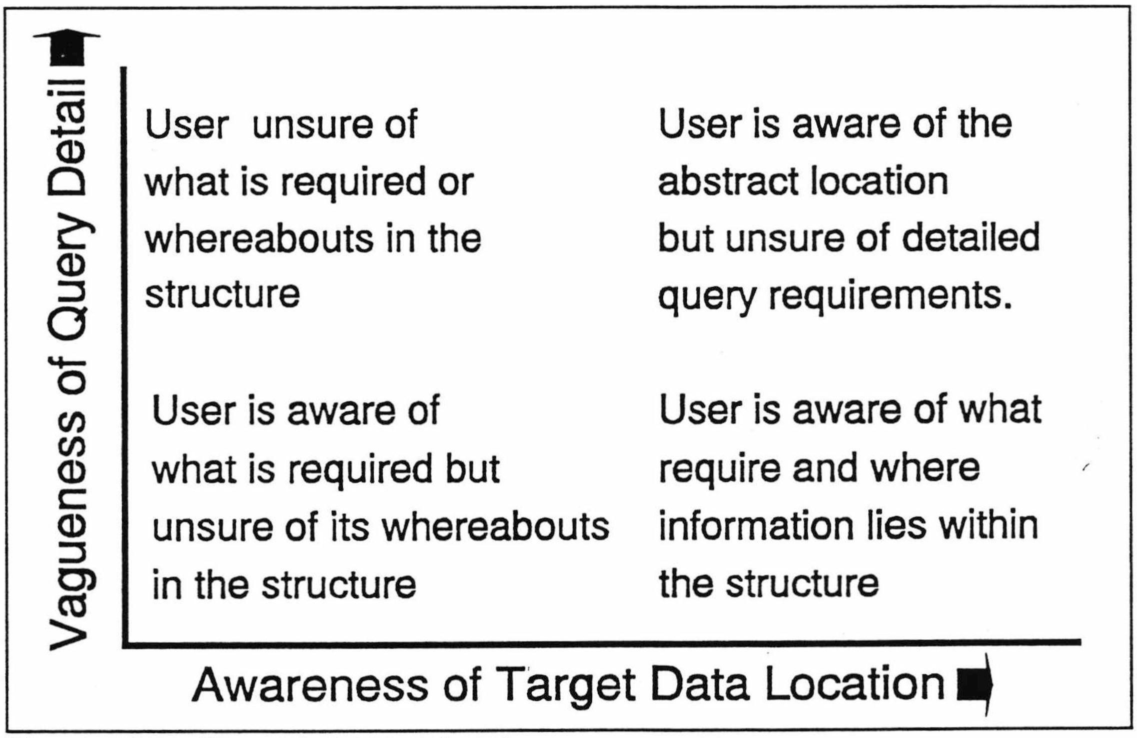
Date[178] provides a second example of subjective definitions, allocating users according to system usage, for example parametric users based on routine interactions, analyst users based as part of job function but non-routine, and casual users that are not task motivated and are therefore still unpredictable.

The allocation of users to groupings based on this analysis is more objective than the criteria suggested by Shneiderman, in that each user will have a formal or informal but identifiable job specification, and hence computer interaction specification. This task specification that will be required of the user, will form the basis of the requirements made by the user of the interface, for example those of a doctor versus those of an accountant.

Each of these approaches has focused on a single aspect of user interaction, resulting in models that are relatively coarse in nature, for example definitional problems of a casual user built on an assumed application. This work presents a process model of user interaction, linking the user and the application through the query generation task to be accomplished.

Two key variables are identified, firstly chapter three identified a key variable in information resource utilisation - the complexity of the database application structure. Regardless of system usage, computer experience or the particular task at hand the user has to interact at some stage with the structure of the application. Secondly the ability of the user to express query detail, for example selection conditions and identification of target data, especially when most users are unaware of what they exactly want in the first place[179]. This expression of what is required has been modelled previously, but only in surrogate form, for example model criteria based on computer experience or mandatory usage implies the user is aware in detail of what is required. However as contactability with non computer literate users increase, this ability of the user to express what is required in application terms becomes less likely, a consideration that should be reflected in user interface provision.

Presented here is a four component model of user interaction, specifically concerning interaction with a database application structure. Each component is part of a two dimensional matrix based on user ability to state query detail explicitly, that is an index of 'vagueness' set against whether the user is explicitly aware of the target data location within the application structure.



User unsure of what is required or whereabouts in the structure	User is aware of the abstract location but unsure of detailed query requirements.
User is aware of what is required but unsure of its whereabouts in the structure	User is aware of what require and where information lies within the structure

Figure 22 User Model Developed

In the north west quadrant lies the user who has only a vague idea of what is required and of where the target data may be located within the application structure. This vagueness confronting the user has to be expressed into the domain specificity imposed by the database structural model. The role of the user interface is then to map this from the vague description defined in the user's conceptual interpretation into the specific application model. This functionality of the user interface can be expressed as a continuum from the command line environment in which the user has to carry out the mapping explicitly, towards the interface environment providing methods of abstraction and representation of conceptual schema.

In the south west quadrant lies the user who can identify what is required, but is unable to specify in sufficient detail the exact location of the query target list within the application structure. This form of user profile is increasing as computer appreciation rises, but where the user remains unable to handle explicitly the complexity of the application structure. The role of the interface is to provide a mechanism to express what is known by the user, building the location specific information as a byproduct relative to what is expressed.

In the north east quadrant lies the user with application structure expertise in abstract terms, perhaps acquired through past interactions, but remains unable to specify the information request in sufficient detail. The role of the interface for this user is to provide query formulation in an abstract environment, insulating the user from the direct application structure.

Lastly in the south east quadrant lies the user who is both aware of exactly what is required and where in the structure the query information lies. The role of the interface is based on transparency, providing the shortest path to query specification, minimising the barrier of the user interface mechanism.

The aim has been to capture user interaction with a complex information structure in a pragmatic form. The model has not concentrated on specific individual user characteristics, instead targeting a combination of user and application characteristics, mediated by the task to be performed. The model therefore provides an abstract design guide for the detailed design and project specification carried out in Chapters Five and Six.

4.5 Choice of Interface Environment

This chapter identified the importance of human factors in any application. HCI has been described as a two component combination, the environment provided and the dialogue performed. Three forms of environment were discussed: linear, non-linear and natural language with the forms of dialogue provided, for example direct manipulation and metaphor.

HCI design is often coupled to user modelling, targeting a specific design towards a detailed user characteristic model. In this chapter a user process model combining both user and application character was described, focusing particularly on the application structure in which the task has to take place. The choice of interface environment for this work has reflected this developed model focusing on a two dimensional graphical environment, coupled with a direct manipulation dialogue. The underlying aim in this HCI construction has been to tackle the accessibility issues raised in chapter Three. The practical expression of this has been to minimise the language of description, focusing on a world of action metaphor. This choice of human computer interaction forms the basis for the prototype developed in Chapters Five and Six.

CHAPTER FIVE

Prototype Initial Principles and Concepts

Contents

5.0 Initial Principles and Concepts

This chapter introduces a graphically driven prototype to address the accessibility issues to a database information resource which were raised in chapter Three. In particular, the idea that the effectiveness of the user interface to a complex information structure can be improved by capturing and providing information from the semantic modelling process is introduced. Aids to user database query are discussed, the latest of which is the use of this semantic information. A prototype based on this style of user interaction is presented in the context of the complex data structure problem and other work in this area.

5.1 Background

Database technology emerged towards the end of the 1960's in response to the growing demand for a single organisationwide perspective of data. Coinciding with hardware advances enabling a shift away from sequential tape based processing towards random access disk storage, manipulation of large volumes of data within an acceptable time frame became possible.

With the new technology came a new perception of data. The previous flat file approach provided a simple way to access and store data. Data was arranged into tables with each row corresponding to a record and each table a file. This led to duplication both within a table, for example through repeated groups and between files, for example the same data repeated for different applications. To enable the most efficient use by the application, a close coupling of code to the file structure was required. In doing so, the development and change of either an application or file structure was slow, for example, ad hoc request generation required additional or rewritten application code. In summary the overall view of stored data is composed of multiple but individual application programs making this approach good for routine day to day transaction processing which uses a predefined part of the overall data structure, but unworkable for generalised, usually one off, requests incorporating the overall structure.

The database paradigm allows separation of this close coupling, through the provision of a single 'conceptual file' maintained by special purpose software[180]. This conceptual file is composed of many physical files, descriptions and indexes often referred to, in total as the *database*. The intermediary function of maintaining this structure is performed by a management system(DBMS) providing a form of data independence, insulating the user from possible change in the physical storage structure. This approach provides an opportunity to model conceptual understanding of the structure, isolating it from the physical implementation detail. The expense of providing such functionality is performance, cost, rigidity and potential inconsistency.

5.2 Conceptual Modelling

Considerable research interest was generated by the publication in 1978 of the ANSI/SPARC[181] three level data modelling hierarchy. Illustrated in Figure

23 this paradigm provides multiple external or user views, a conceptual or logical model of the complete database and an internal or physical storage model.

At the top level, the external user models are presented. These provide a subset of the overall conceptual model as viewed by a particular user group or application. This view is provided either directly via a query generator or indirectly via an application program with embedded query calls. The conceptual model provides the complete representation of the information content from which these external models are derived. An external model is therefore an abstraction or projection further away from the complete conceptual model, directed at a particular application.

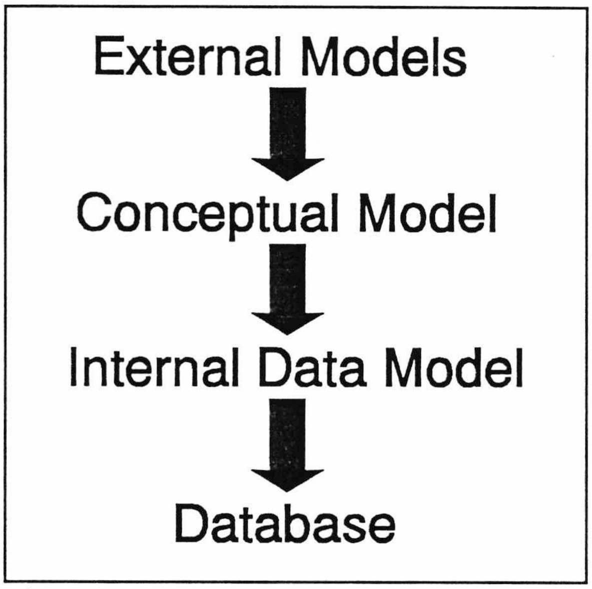


Figure 23 ANSI/SPARC Three Level Data Model Diagram

The conceptual model provides the bridge between the user and the database designer or modeller. The term conceptual model can be split into two sections: the *generalised* conceptual model and a *DBMS specific* data model. The data modeller constructs the generalised conceptual model by investigation and interpretation of the surrounding real world environment, using tools such as data diagrams, inventories and data dictionaries. It is rich in real world semantic detail, often being expressed in a graphical notation, for example Bachmann diagrams, regarding the data structure or ER diagrams for the derivation of an enterprise schema.[182]

Chen gives the most common and still influential approach to conceptual modelling - the Entity-Relationship Methodology(ERM). The ERM offers a three component classification of real world objects: entities are 'things which can be distinctly identified', a relationship is 'an association among entities' referenced by degree (one to one, one to many and many to many) and attributes or 'values' are 'objects which serve to characterise either a relationship or an entity'. In so doing a broad generalisation is provided whilst retaining a mapping to the more specific database schema.

At this stage in the modelling process, this rich semantic information must be mapped into a more constrained DBMS specific conceptual model. Such a mapping usually takes the form of a data description language(DDL) in which it is difficult to express much of the semantic information derived from the more wide data investigation and analysis stages of the modelling process. Usually such a mapping is limited to the expression of normalised relations, for example in relational and network models relations between entities are assumed to be one to many with one to one and many to many relationships treated as special cases. Much semantic based information is therefore lost in this conversion process of squeezing the rich real world semantics into the rigid DDL, information that would prove vital to the user interpretation and manipulation of the information contained within the database application.

The DBMS specific conceptual model forms the basis for both the further design and construction of the internal database model, as well as providing an overall framework in which the user has to formulate and specify requests for information. Three classical[183] specific models exist: the network, relational and hierarchial. These will not be discussed in detail here, see Tsichritzis and Lochovsky[184] for further discussion.

5.3 User Aids to Database Interrogation and Interpretation

The purpose of an interface (as discussed in Chapter Four), is to successfully bridge both the gulf of execution and evaluation which separate the user and application, maximising directness[112] by drawing the user into the system as close as possible. Aids to this interaction process can be traced through three stages, paralleling development in database technology[185,186]. Initially at stage one the networked and hierarchial systems were prevalent, offering the first complete packaged approaches to the database paradigm. Based predominantly on record manipulation, the two key players in the field were IDMS (CODASYL)[187] and IMS (IBM)[188]. User interaction was within a procedural command line environment. The user was required to manipulate the logical structure directly, specifying requests in one attempt within an environment with little help or feedback mechanisms, for example the ICL

PAS introduced in Chapter Three. The role of intermediaries developed quickly to soften the harsh computer environment from the comparatively novice user, for example the development of an Information Department in all Districts discussed in Chapter Three.

During the 1980's these systems were largely surpassed by relational systems. For these second generation systems, the main advance was through increasing semantic content, the real world was not always hierarchial or networked in nature but a mixture of the two that could now be reflected in the data model itself. From the user viewpoint, two forms of interface style developed. Initially the use of forms dominated database querying and manipulation. This is still the case in the personal computer market today, for example REFLEX(Borland) and HELEX(Odessa Co) for the Apple MAC, or DBASE(Ashton-Tate) and INGRES(Ingres) in the IBM PC environments. Full two dimensional use was made of the screen, but the functionality provided followed closely a linear path, often remaining predominantly textual in nature, for example browsing systems such as the Dynabook Project. Database accessibility improved considerably, for example the initial and best known query environment supporting two dimensional visual representation was QBE[189]. CUPID[190] provided the same functionality as QBE, but is based on an iconic approach rather than the tabular approach of QBE. TIMBRE[191] was perhaps the most advanced interface environment at that time offering window based browsing over the relational model.

The increasing spread of workstation and main frame graphics capability facilitated development of the non procedural direct manipulation interface style of user interaction, for example Rooms[192]. Initially this style of work remained closely tied to the underlying logical model of the database structure. However during the 1980's, increasing use was made of Entity Relationship(ER) diagrams for schema definition or database query. Provided with only limited graphical capability these systems began to move away from the rigid computer based interpretation of the database structure, towards a more semantic interpretation. Chan and Lochovsky[193] provided the first attempt at introducing graphical Entity Relationship diagrams as a basic design function to database modelling definition. The advantage of full screen representation of an abstract model for query specification or database query was beginning to be recognised. However this was implemented with weak direct manipulation and functional decomposition, the full potential of the graphical interface remained unemployed. GAMBIT[194] focused on strong functional decomposition, but again remained coupled to a weak direct manipulation environment, buttressed however by extensive textual guidance.

GUIDE[195] was the first application to specifically focus on the control and manipulation of the graph view. It was quickly realised that as the semantic diagram increased in size, such a graph became impractical to search manually. GUIDE provided a menu structured system, based on a subject hierarchy coupled to individual screen nodes being allocated an 'importance factor' ranking. An inbuilt presentation abstraction mechanism was therefore provided, with the user able to express the degree of detail required for each query. Detail ranking was applied by the systems designer, but which may be at variance with any particular user interpretation. However at the very least, GUIDE recognised graphical structure presentation and manipulation as of prime importance to encouraging schema browsing and overall user help in query specification.

During the late 1980's and 1990's there has been movement towards third generation database systems, away from business based applications that have so far dominated database development. Coupled with this is a new approach to user interaction style, focusing towards user interaction with the conceptual structure of the overall database, rather than towards its confined logical structure. This has the key advantage of lowering user mental load in query specification, 'softening up' the system.

With this new approach it has been recognised that the generalised conceptual model established by the systems designer represents significant user interface potential. Specifically there has been an attempt to link database schema design and database development with an environment for database query by the final end user.

SKI[196] was the first interactive schema development environment based on a large and possibly confusing data model - Sembase. SKI is composed of a panel style interface with system objects represented by various symbols.

Based on a highly structured view of the schema, SKI provides close panel control with several methods for deriving partial views of the complete graph. Similar in approach to GUIDE, SKI focuses upon the user definition of initial construction points of the query from which an incremental graph of data relationships is built. SKI however cannot provide a global view of data in the structure.

ISIS(Interface for a Semantic System)[197] can support a global view of data, supporting two schema viewing levels, an *inheritance forest view* and a *data view*. ISIS provides a sophisticated mouse based editor environment combining database construction, browsing and query specification based on the Semantic Data Model (SDM)[198]. As in SKI, query specification is performed through subclassing or grouping classes from the inheritance forest into a third view - the *predicate worksheet*. The final product is a new class generated and installed into the inheritance forest. Such an approach of separating the overall from an additional component suits handling complex schema well, but its use of patterned icon representation is confusing.

SNAP[199], unlike SKI or ISIS, is based on an extensive semantic model (IFO)[200] providing schema representation for flexible rearrangement based on *fragment representations*. Based on a workstation environment, the user is able to move, size and delete screen objects to generate a query graph in an answer window. Limited abstraction mechanisms are provided, principally a toggle mechanism for removal of sublinkages, but the user remains exposed to the complete structure each time to formulate a query. A principle restriction of SNAP is that any knowledge acquired by the user during query specification, for example about the structure of that local segment related to the query itself, is not capitalised back into the structure to aid future query specification. Each new query is started with a 'clean' structure.

This past work indicates that there is a general danger of heading away from 'user participation', for example providing screen structures defined in a systems manner or where the user is unable to define own objects into the screen structure. Prima facie ISIS provides a form of user participation, allowing the screen structure to be manipulated directly by the user, with new queries added to part of the class and inheritance structure. Such representation remains however very close to the programmer interpretation of the structure, for example through 'subclassing' and linear query expression that may not necessarily coincide with a final user interpretation.

In addition to this problem, for each approach an underlying danger remains, replacing the mental load of interaction with the detailed logical structure with the mental load of interaction with the new semantic structure. For example size and complexity, but also unfamiliarity of the semantic graphs remain prevalent, requiring the user to take on the role of data modeller as well as data manipulator.

SNAP and ISIS both provide examples of this problem: In SNAP there are many symbols representing a mixture of available types of object (eg printable and abstract) based on a complex semantic model, whilst in ISIS there are many forms of icon shading and hatching. A possibly confusing display results with many forms of symbol, some repeated, for example no distinct representation is maintained in SNAP.

The most recent work in this area has been QBD[201]. This is based on the entity relationship model as underlying semantic schema, providing a powerful recursive query function. The general approach to query formulation is based upon the identification of a concept from which a semantic path is traced to all parts of the structure of the resulting query. It is unlikely that real users in the real world formulate a query in this manner. However when complete, the results from this development process are then discarded, so along with SNAP the user starts with a clean slate each time. Suggested here is a system which provides graphical query specification within a simple semantic data model in any order that the user finds advantageous, for example incremental building on past queries. The aim is to promote user participation minimising the language of expression with a small set of symbols representing the world of action for the application. The question of pathing and query navigation is relegated to a final issue once the main determinants of the query are established.

5.4 A New Approach

To address the issues identified in Chapter Three and using the tools and user model discussed in Chapter Four, presented in this thesis is Query By Concern(QBC). This section will introduce the ideas and key issues behind its inception. Chapter Six will provide full implementation detail and a worked example.

The aim of this effort has been to build on the work of ISIS, SNAP and QBD in providing the final end user with a dynamic environment in which can be incorporated both personal and organisational information as a basis for task and query formulation of the information resource. Following QBD but unlike ISIS or SNAP, this work is based on the Entity Relationship semantic model coupled with the direct manipulation style of user interface environment. The prototype design offers principles general to all data modelling paradigms. QBC as presented is however demonstrated exclusively with the network data model providing uniformity with the NHS DIS database structure presented in chapter three.

5.4.1 Key Principles Behind QBC

Several areas of weakness in the related work parallel the key issues arising from Chapter Three and the field study.

Firstly the requirement for the user to specify the query within the complete semantic network each time, for example limited methods of abstraction were provided within the semantic structure, exposing the user to the complete structural/dynamic complexity each time (ISIS and SKI). From the field study, for each query formulation of DIS, the user was exposed to the complete structure of the database, even if parts were never used nor related to that District, for example for inclusion of Regional abstract data sets.

Secondly the lack of emphasis towards tailoring this semantic view towards a personal or organisational interpretation of which individual task interpretations are key components (SNAP). Similarly from the field study the data model provided was a fixed, past interpretation of the information resource, produced by a systems analyst (that is non end user).

Thirdly, different users have different perceptions of the information content of the data resource. Providing an environment based on a single semantic model does not address this issue. Different users as well as the tasks they perform, require different semantic models accessible through the same interface environment, for example a notion of information context is required[202].

Lastly SNAP, QBD and SIG[203] provide an emphasis towards one off query formulation rather than providing an environment in which this semantic model develops in parallel with user interpretation. QueryMaster provided a single, rigid user interface environment with no facility for successive query refinement.

This work has aimed to address these issues in a practical manner, particularly those issues identified from the practical work. An environment specifically aimed at providing not only user and organisation based interpretation, but also localised task based interpretation of the semantic database structure is provided. In this manner, a dynamic semantic structure, maintained onscreen, which will parallel user changes in interpretation of the structure of information held is aimed for.

QBC provides a dynamic interpretation of the information structure within the database in which personal, organisational and task based information are aggregated. ISIS is in theory the closest to this work, providing additional class structure for each query generated that is then linked into the semantic structure. SNAP and QBD both provide a query specification engine, but provide no ongoing development of semantic interpretation of the information within the application. The work here is aimed at building an environment which will parallel user development of knowledge about the system, something which no previous approach has explicitly attempted.

The basis of this argument is that each user approaches a computer system with an expectation coupled to an internal mental model. This model has then to be explicitly defined and converted into the conceptual model provided by the system for each user query (the gulf of evaluation discussed in Chapter Four). The user mental model will also change with each approach to the information system, being based on a personal interpretation within a personal environment. External situational variables will also change with each request. These can never be predicted or quantified in any meaningful way before the user interaction, especially either because the accumulated knowledge of the previous request has now been added into the user's own knowledge library, the circumstances may have changed or because the user now has a detailed interpretation of another area of the structure.

Such a mental model can be split into two distinct areas:

- A base interpretation of what the information system as a complete entity can provide the user, for example "patient information". This view is based on a mixture of personal interpretation, rumour, user background and past interaction history with the information system. Such an interpretation provides the general backdrop into which more detailed local level views can be placed in context.

- A local, detailed level of interpretation of the task, based on particular detailed user knowledge of the particular task to be handled. Such a task may involve different "areas of interest" within the database that can be grouped into user areas of "concern". Such a user model may vary from a detailed appreciation for what is required, to a vague notion of an area of interest that the user wishes to specify.

The first task of any interaction with a computer based information system is to integrate these two areas of knowledge into a mental model of the current state of what the system has to offer for the current task. This mental model has then to be interpreted by the user into the conceptual model as specified by the systems programmer. This interpretation has then to be expressed in a computer programming language ready for submission to the database internal model.

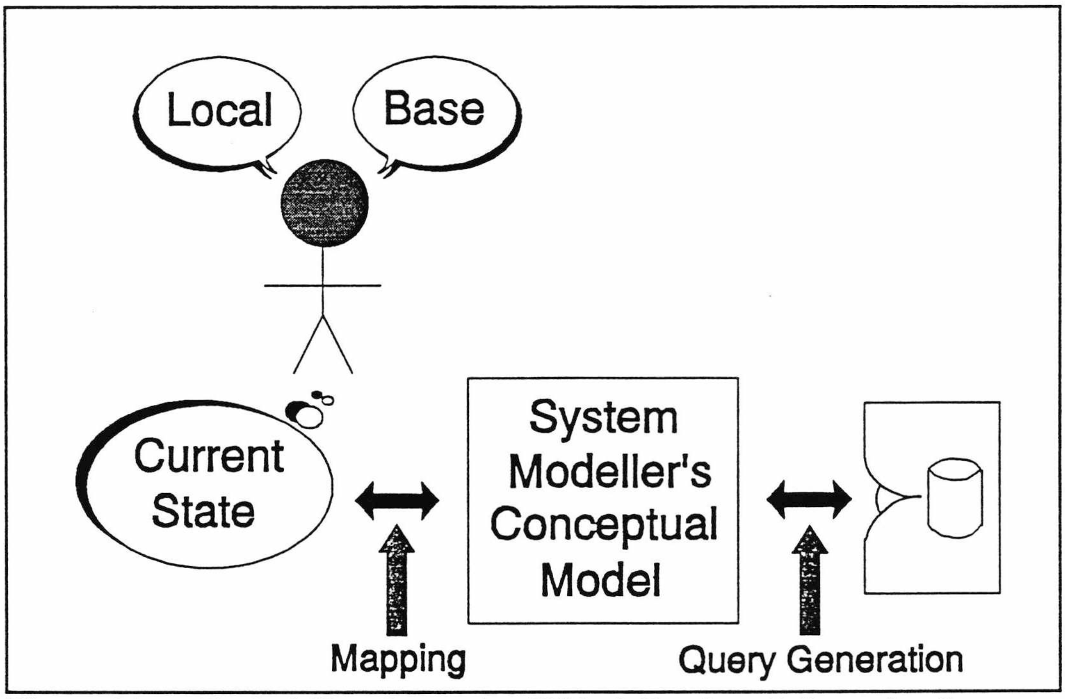


Figure 24 Typical User Interaction With A Database Information System

To reduce the mental load of these conversion processes, this work is to provide a user interpretation environment in which what is required for the current task is specified explicitly in terms of an interpretation of the information system at the current time. This personal interpretation is to be built up over a period of time from a base world interpretation, provided on first contact with the system. This base may be provided on the basis of job function, access permissions or a standard interpretation, but acts only as a starting point. As the user interacts with the information system, for example by specifying information requests, the user grows in awareness of the information system itself, but also the information contained within it. The current request is specified in terms of the user's current interpretation and understanding of the information within the system at that moment. This interpretation knowledge is often buried and lost to an internal library of macros and possible documentation, but forms a vital part of the conversion process outlined above. Such knowledge should be capitalised into the system to aid future interpretation.

With the addition of each subsequent task, the interpretative status of the user will change in two ways. Firstly knowledge about the system itself will increase, for example detailed information will be gained about a particular section of the available structure, increasing the familiarity about the structure of the information itself. Secondly an increased awareness of the knowledge held within the system itself will have been achieved. The user will have a better understanding of what is available in that topic area and what can be asked for again. It is considered in his work that the task of the interface is to represent and incorporate this knowledge into the overall user interpretation.

The aim is to make the structure that is to represent the semantic content of the database application sufficiently abstract to facilitate this, that is by using the generalised conceptual model captured from the systems designer as the base start up view. By so doing, the user can now manipulate this representation in terms of the processes described above, both initially at an abstract level and subsequently at a more detailed level for query specification. The prototype will therefore need to meet the following requirements:

- The capture and manipulation of the database semantic (generalised and specific conceptual) models from the systems designer with DDL code generated as a byproduct.
- The presentation of a generalised conceptual model to any end user as a base start up view upon which localised personal interpretation is built.
- The provision of an environment structure encouraging the saving, reuse/capitalisation of user tasks and interpretations of specific structure areas in a graphical context.
- The provision of graphical based tools for the end user to tailor the semantic structure towards user, organisational and a task oriented interpretation.

To achieve these intentions, the application should be considered in two closely linked halves: the systems designers perspective in which the semantic and DDL code are specified, and the query development and interpretation environment for the end user.

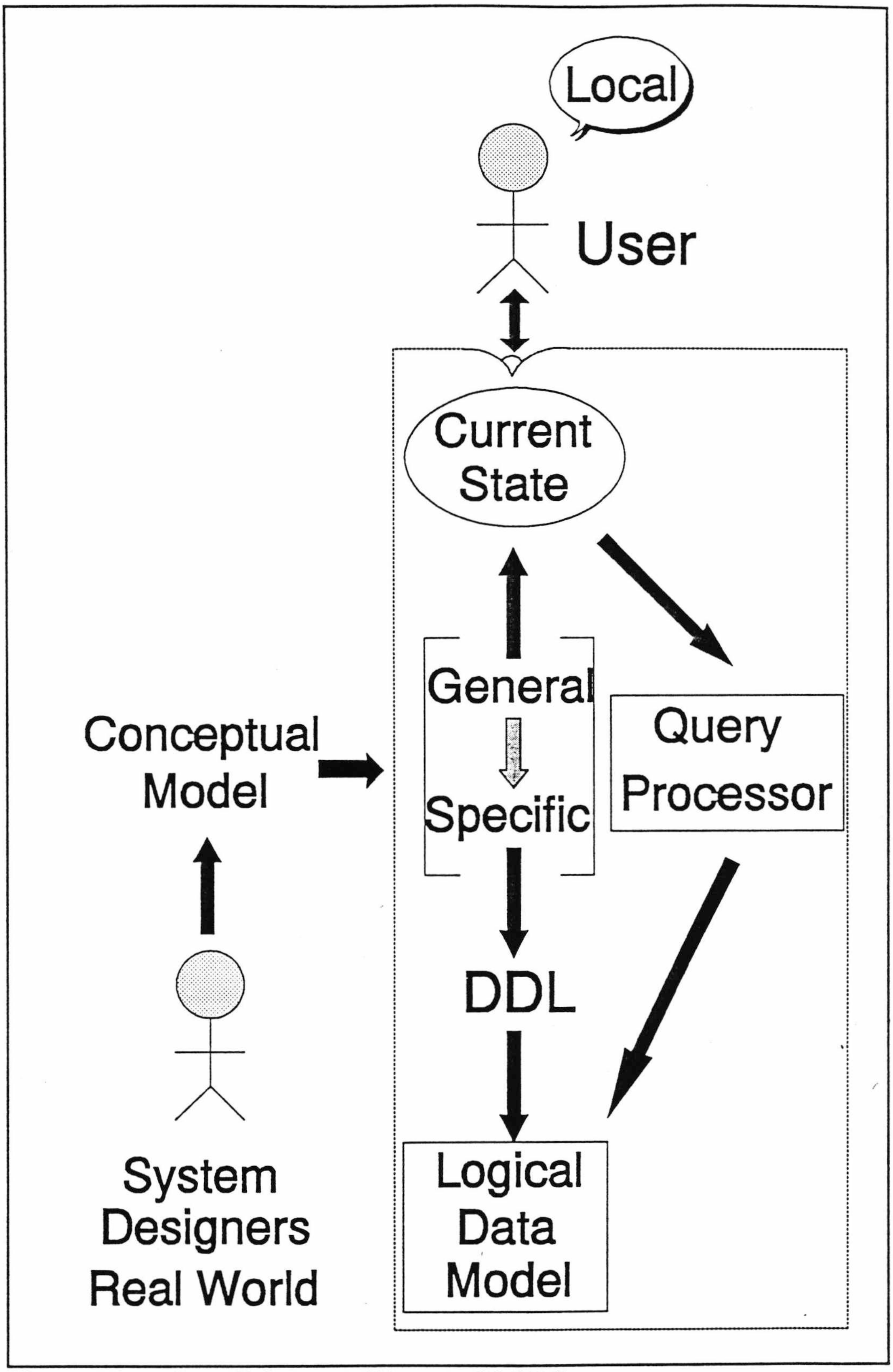


Figure 25 Overall Structure Diagram Of QBC

5.4.2 Systems Designer Structure

The emphasis in this environment will be to provide three closely coupled views in which the system modeller is to express the database structure in graphical terms. The three views are:

- Generalised view access in which a high level of abstraction is provided based on what clusters or groups of entities and relationships mean within the application setting. Clusters or 'areas of interest' provide a mechanism for grouping together those entities which 'naturally' occur together within the application domain. An environment is provided in which the modeller can develop the generalised conceptual model.
- Specific view access in which entities and relationships are directly expressed in a graphical format. This view provides the graphical representation of the specific conceptual model with implicit mapping to the generalised view.
- Direct access via a code browser in which raw DDL code is provided as a byproduct of the graphical specifications made in other views.

Splitting the views in this manner will provide the maximum flexibility to the data modeller, providing either a top down or bottom up modelling approach. For example the top down approach would allow working from the generalised view dealing in abstractions to the detailed DBMS view, whereas the bottom up approach would allow working from the DBMS specific view to the complete abstraction summary. Output to the code view would be generated automatically, based on the graphical actions of the database designer within the other two views, *not* within the generalised view. The aim of the generalised view is to provide a modelling tool to the data modeller that will then be directly relayed onto the final end user as an interpretative base point in query formulation. Comments, descriptions and notes will be provided at any level, provided in the code view for internal system design, at either the generalised or DBMS specific views for the end user design.

The generalised view allows the modeller to specify in abstract terms the interrelationships between clusters/areas of interest of entities, attributes and links in a semantic fashion, without explicit regard to the constraints of a mapping data description language. The whole screen is available to the modeller as a workbench to describe the application to be modelled. The modeller will be provided with graphical tools for specifying screen objects as required, entities, attributes and relations, and for manipulating their presentation, for example moving, hiding and shaping. Once achieved it is the modellers task to map this into the DBMS specific view, and subsequently into the DDL code view. The main advantage in taking this approach is that the complete generalised view will form the basis of the final end user interpretation of the database. The final end user would then specify a request in terms of the interpretation of this generalised view and the modeller mapping will be generated automatically by the application.

The DBMS specific view provides DBMS representation of the conceptual information portrayed in the generalised view. This view forms the actual mapping to the DDL for the code view. Here the data modeller has to develop entities, attributes and intermediary links. The aim with this view is for the data modeller to develop a DBMS specific model in a graphical format that will then form the second stage of the end user interface. The modeller will be provided with graphical tools for specifying screen objects - entities, attributes and linkages and for manipulating their presentation, for example moving, hiding and shaping. The consequences of these actions will be produced in skeleton DDL in the code view.

The code view is provided as a browser over the skeleton code derived from the modeller's actions in the DBMS specific view. The syntax of the DDL grammar can be disaggregated down into elements and properties. Providing a browsing function is superior over a simple form fill approach in that most elements are repeated groups with many properties, flags and switches. Browsing through a standard structure enables fast and effective location and manipulation of the code being generated. Such a view is required to reinforce to the modeller that satisfactory code is being generated, allowing in addition fine tuning, especially where the graphical operations can only provide part of the detail required of the DDL. The graphical operations could be expanded to perform all/most of the DDL specification but it is very likely that this would overload the functionality of the graphical medium. Additionally, computer based personnel may prefer to specify some parts in direct code rather than the perceived slower WIMPS environment. This approach combines both, the modeller can follow any personal methodology required or style of interaction to complete the mapping process.

The overall design approach is therefore tailorable directly to the personal preferences of the modeller, whilst as a byproduct more semantic information is relayed onto the end user as an interpretative aid to query specification. Other possible approaches of straight browsing or simple form filling do not provide the capture of the generalised view for the end user interpretation, an essential key to the improved interpretative ability of that interface design.

5.4.3 End User Structure

The emphasis here has been to build the prototype structure around the form of mental user interaction identified previously. Two closely linked levels of interaction are provided. A base start up view provides general interpretation of the structure of the data held within the application and a 'concern' view provides the incremental information related specifically to a task or refined group of interest areas.

The base view is the generalised conceptual view captured from the systems designer. This will act only as a startup/initial contact interpretation for the user with the system on the basis that this conceptual model provides historical interpretation provided by the systems modeller (who may know little about the information content or the tasks to which the information is to be put of the database structure being modelled). Such a view if taken in isolation will be historic in nature, providing a third party to the database querying process. The aim here is to provide this as a base upon which the user builds, rather like the progression from the formal to informal organisation chart. The formal chart provides the basis for interpretation of an organisation structure, rather than by starting with trial and error to establish a personal interpretation of the organisation structure.

The user is provided with this conceptual model and the graphical tools with which to tailor this towards an overall personal interpretation of the structure of the information resource, for example move, hiding and shaping. The view is abstract in nature, providing the opportunity for user interaction in a vague manner, selecting areas of interest in broad terms that the query may cover.

Many task based systems require the user to be able to specify the complete query in one session, an approach close to the GOMS model[119]. It is however unlikely for the user to have all the required information to hand to provide a complete specification, graphical or not. A more likely process as identified in Chapter Three, is for the user to aggregate information about a particular problem together reaching a point at which enough detail is available to submit the query.

This work provides the notion of a concern[204], a separate confined problem domain, specified from within the base structure itself. This is initially specified in terms of the areas of interest that relate to the query plus other additional information as available to the user. A concern may exist beforehand and/or after a query is generated. Its purpose is not exclusively query generation. It may act as a holder or restatement of several areas of interest, providing a mechanism for aggregating together areas of interest into a new area of interest. Following ISIS, a new subject in the menu structure is created. However if used for query specification, a concern provides the mechanism to specify a query within the context of the overall semantic interpretation of the application.

On opening up a concern, the selected areas of interest are translated into the corresponding areas within the specific conceptual model, aggregated together and loaded into the newly opened view. New links are added by QBC as required to maintain the semantic correctness of the data model. This provides a subschema of the areas of interest the query is to cover. The user is provided with graphical tools to tailor this subschema towards a personal interpretation of either the task at hand or into any alternative graphical interpretation. This subschema represents a localised interpretation against which a query may be specified or simple refinements of part of the base structure, tailoring the base view for future use and possible query. In this manner the additional knowledge derived in generating a query is added back/capitalised in context into the base structure rather than being lost to an obscure macro library.

5.5 Graphical Properties of the Work:

There are four main properties of this work warranting further discussion: the general and semantic level graphical display principles used, the use of concerns, the maintenance of a path/navigation for query processing and screen object basics. Each will be considered here in turn.

5.5.1 General and Semantic Detail

At a general level, a basic overall view structuring policy has been adhered to. There are several examples of this: a common way of managing overlapping windows is followed, of placement and construction of menu options, the initial choice between menus and buttons and the confinement of menu options to common functionality across views. Work from psychology and HCI has identified the use of modes to be confusing, incurring additional mental load upon the user. Postfix processing in a direct manipulation environment has been successfully used to avoid this, for example screen objects are firstly identified, with the operation to be performed requested postfix. In addition, to maximise the benefits of the direct manipulation interface style, off screen structures that would be out of reach of the user have been minimised.

At a more specific level, the basis of the work has been user interaction and expression of semantic structure within a graphical context. From a theoretical position the Entity Relationship paradigm provides a simple and yet powerful formalism that can be mapped into any logical data model. At the same time it remains conceptually simple enough to be expressed graphically, enabling end user development of semantic interpretation.

Work has been done to extend this paradigm to include a wider semantic base, including type and existence characteristics[205,206] and most recently further integrity constraints[207]. The work discussed here has not attempted to include such extensions, as from the user's perspective it is often unnecessary and confusing, both directly for the end user and indirectly for the display. It may be of benefit to the systems modeller, providing a richer modelling environment, but for the end user it has been omitted. A further example is provided in the representation of link relationships for the end user. These are provided on the basis that further descriptive characteristics (for example existence and degree) would clutter and confuse the display, causing confusion with more symbols to comprehend and successfully manipulate. A binary representation has been adhered to, concentrating on the simple existence or non existence of a relationship[208].

5.5.2 Graphical User Concerns

For the end user to specify a query, the main form of interaction is through the development of a concern. Part of the menu functionality required is to be able to combine past concerns, possibly with additional areas of interest. This combination is performed on a graphical basis, aggregating the corresponding specific model areas associated with each abstract object selected.

In performing this process, a particularly graphical approach has been taken. Each abstract area of interest holds a collection of base list objects from the specific conceptual view that are to be mapped into the new concern. Considerable work from the graphics field has been undertaken in this area of graph construction, for example algorithms to minimise screen clashing and line crossing. Rather than reimplement parts of this work, the approach taken here has been to capitalise on the screen placement information already captured within the application from the systems designer. The base list associated with each area of interest is regarded as a tile with any links exiting the tile considered adjustable. Objects that appear in more than one tile weld the parent tiles together into a single moveable tile.

The tiles are then placed on the view and 'bubbled' towards the origin. This provides a simple but effective approach to view construction maintaining as far as possible the spatial arrangement of the semantic structure as specified by the systems designer. Once the user is satisfied with the pictorial representation of the concern, this localised interpretation can then be integrated with the overall user interpretation of the information resource by collapsing the concern to a single screen object and inclusion in the base view as any other screen entity.

The key problem with this approach is the decision of which links to display to provide a complete semantic picture of the chosen concerns and areas of interest. In theory there is a link from every node to every other node.

Providing all such links individually would swamp the graphical display and the user, minimising the value of presentation. Two key ways of filtering these links are used. Firstly only those links from border nodes are considered adjustable. Border nodes are those nodes that have an attached link passing to a node outside the current tile. Secondly the links themselves are expressed for the graphical representation in terms of the generalised conceptual model, that is the areas of interest through which they pass. This reduces the number of links required, aggregating those links that pass though the same conceptual path into a single link.

5.5.3 Pathing/Navigation

In providing an environment in which the user manipulates screen objects directly within the Entity Relationship paradigm, there is scope for the capture of selection detail specification into the individual screen objects themselves. Processing the overall query may then be performed on an incremental basis, as the user interacts with screen objects (for example as in ISIS and SNAP). Integrity checking can be performed at the local level as each selection specification is implemented, providing scope for an immediate local contextual error response. A complete integrity check will be required postfix once the user is satisfied with the pictorial representation of the query.

A key difficulty with using the semantic data structure as perceived by the user for query formulation, is the necessary maintenance of a satisfactory path for the query to follow through the logical data structure. Different paths through the logical structure may generate different information through incremental database operations. To approach this problem, a simple user rule has been imposed: keep all screen objects linked on screen that are to form part of the query. In doing so, this will preserve some semantic value in the structure (for example they should be linked because they form part of the same concern query at least) and provide an indication to the application of the path to be followed. This problem arises in particular during generation of a concern by the application and when this concern is processed into a query. To build a concern each selected area of interest is linked with reference to existing links in the specific data model. These links are aggregated where possible and to maintain semantic presentation, are depicted in terms of the conceptual model. Each macrolink therefore holds a mapping to the specific data model, information which is required when the concern is processed into a query.

One of the primary aims of the work has been to provide a graphical approach to query specification, insulating the user from interaction with the logical structure of the database. However with any graphical structure, it is very likely to have multiple paths connecting nodes, introducing the possibility of variant information retrieval. For this problem, unless some form of artificial intelligence (AI) is introduced into the processing, some form of user intervention is required to identify the semantically correct path in terms of the real world request. This may be performed textually but following the general philosophy of this work, the graphical approach has been implemented. If a choice is required, the user is requested to identify screen objects via which the query should be routed rather than the complete path, removing the need for each possible path to be redrawn. Presenting each possible path separately would incur confusion and a mental load, drawing the possibilities out of context of the general structure.

5.5.4 Screen Object Principles

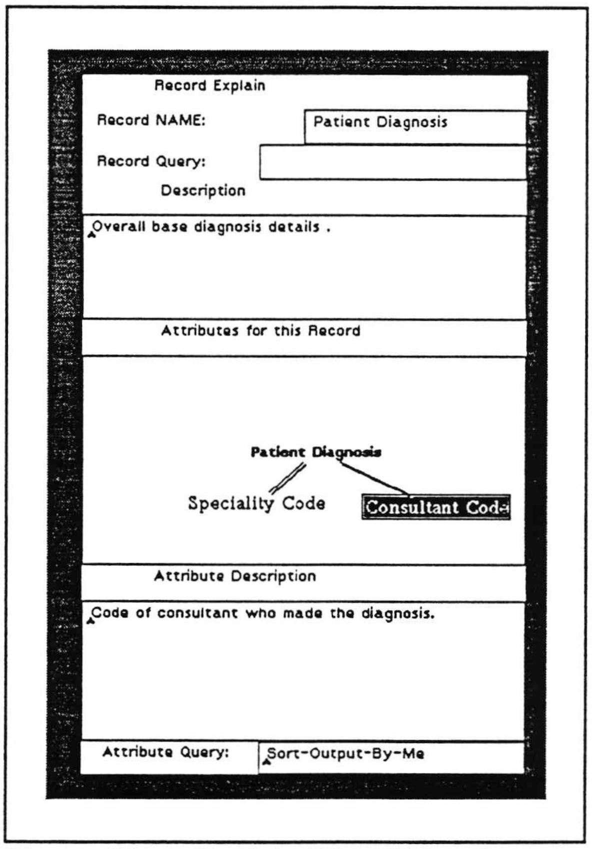
All items that are presented to the user are to be screen objects. These are objects that occur 'naturally' within the application domain. They will therefore include complete concerns (whether collapsed or expanded), entities, attributes and relationships (from the Entity Relationship model) as well as any screen buttons or menus. A group of fundamental operations is to be provided for every screen object, to be individually interpreted by the screen object itself (polymorphism). These operations provide that every screen object is to be:

Selectable	Operations are to be performed post fix on screen objects already selected
Movable	Around the screen to provide spatial association and user presentation control
Removable	From the screen completely
Replaceable	Recovering from a buffer any removed screen object
Aggregatable	Aggregate several screen objects into a single representation
Changeable	Title or screen appearance

Expandable Interrogated to establish contents (attributes or concerns) or selection conditions applied or internal objects of an aggregated object.

Linkable Any object can be linked to any other object within the screen structure

Explainable Each object carries an explanation of its purpose initially entered by the systems developer. This is then available to the user to be tailored towards personal or task based interpretation. Figure 4 provides an example explain view.



Record Explain

Record NAME: Patient Diagnosis

Record Query:

Description

Overall base diagnosis details .

Attributes for this Record

Attribute Description

Code of consultant who made the diagnosis.

Attribute Query: Sort-Output-By-Me

Figure 26 Explain View Example

In addition to the basic screen objects principally provided by the ER model, two forms of user defined object are provided. In future work this may be expanded to include complete data sets[209] as well as other forms of data, for example orienteering towards Hypermedia graphical and sound objects[210]. The two additional objects are text and link objects. A text object provides a screen object into which descriptions or notes may be directly stored and placed onto any semantic view, linked directly to an existing screen object. A link object provides the ability to infer tailored relationships between any screen object which may or may not immediately exist within the logical database structure. This is a powerful mechanism for the tailoring of semantic structure towards a personal organisational or task based interpretation.

5.6 Summary

This chapter has aimed to justify the use of the abstract conceptual model as a basis for user interaction and database query development. The development of user aids to database query is traced thought three parallel stages to that of the database technology. The third/current stage focuses upon the semantic data model as a basis for user interaction. Previous work in this area is discussed, from which several areas of weakness are identified, particularly the lack of dynamic semantic model change towards personal, organisational and task based characteristics and the minimal abstraction/aggregation mechanisms employed exposing the user to the complete complexity of the semantic structure for each query interaction. A prototype, QBC was introduced and discussed in the light of this previous work concentrating in particular on the use of concerns and areas of interests to describe the semantic structure. The next chapter, chapter Six will discuss this prototype and its implementation in more depth.

CHAPTER SIX
Implementation of QBC

Contents

6.0 Introduction

This chapter will present the detailed design of the developed prototype system. The basic programming paradigm is established together with the choice of programming environment. A detailed analysis of what has been achieved is presented together with a summary of progress towards the issue of accessibility and the prototype requirements set out in Chapter Five.

6.1 Choice of Programming Paradigm

Over the past four or five decades command grammars for instructing computers have evolved through many dialects, often paralleling developments in hardware advances, for example processing power. Progressing from assembly language, PL1, BASIC and others through to high level languages such as Pascal or C, each has had to conform to the same organisational paradigm, providing an emphasis on the flow of control through the application. As an example, the programming languages provided for the first generation of computer hardware were essentially machine code based, with primitive control structures amounting to branching or the GOTO syntax. As computer languages have become more powerful, a greater degree of procedural abstraction has been provided, developing more complex control structures such as IF THEN ELSE or REPEAT UNTIL.

With higher levels of software aggregation, the number of tasks to be programmed and numbers of staff involved increased dramatically, requiring for the sake of practicality a structured logical approach to systems design and coding. The most common approach employed is known as 'structured programming', the 'top down approach' or as 'functional decomposition'.

The functional decomposition paradigm provides a well ordered route to handling the complexity of a software project. The aim is to decompose the structure of an application into smaller and subsequently more manageable robust units. Each unit is allocated a single discrete task, calling other units as appropriate. The root to the resulting tree structure provides the pathing mechanism or 'main program' for the application. The structure of the application is built around the single main calling procedure that ultimately concentrates on the function of the application. The data structure is of secondary importance remaining purely passive, being either lumped together as global to the application or passed around the control structure as required.

In this manner it is assumed that the application can be adequately described by the main central calling procedure. Such a rigid approach does not lend towards either the reusability of code or the evolutionary design of software. It is more likely that the types of data available to the application structure remain static whilst the demands placed on the application of the data will increase, perhaps even during the code development cycle. It has even been suggested that it provides a short term tradeoff in developing software speedily against the long term expense of rigidity and lack of reusability.[211]

Such an approach is often seen as 'natural', but on closer inspection may be viewed as artificial, awkward and conceptually difficult to follow[212]. The key reason for this is that the application is built around the logic and architecture of the computer, rather than around the logic and architecture of the real life problem.

An alternative to this top down approach evolved during the 1970's, stemming from simulation theory. The roots of what is now termed object oriented programming can be traced to the Xerox PARC research labs and Smalltalk-72[213]. Through the combination of Simula-67, turtle graphics and what is now termed the 'Direct Manipulation' style of user interface (as was previously discussed in Chapter Four), the first object oriented software environment was created[214,215]. Since then many variant environments have grown up, for example Objective-C[216], Eiffel[211], Actor and many revisions of the Smalltalk environment itself, now reaching Smalltalk-80 Release 4[217].

6.1.1 The Object Oriented Paradigm

The emphasis behind this approach is data abstraction, rather than the procedural abstraction provided by the top down methodology. No linear streams of code are provided for execution, instead the 'main program' concept is replaced by the paradigm of objects communicating via message passing. In this bottom up approach, rather than concentrating on the functionality an object is supposed to ensure for the application, the object itself is the focus of the developer's attention.

6.1.1.1 Objects, Classes and Messages:

Each object is a self contained complete package of information with a description of its manipulation. An object is composed of two parts: a public and a private part. The public part is the objects interface to the outside world or to other objects in the application. The private part is composed of a combination of data and methods that are executable in response to messages received from the public part. Access to the private part can only be by this mechanism, with objects able to send out further messages to both itself and other objects in the system as a means to instigate the required action. For an object to do anything within the system, an explicit message must be broadcast, to which the receiving object must have a correlating method. The underlying basis is the IF-THEN structure, IF this message matches an expected message type, THEN execute the associated routine, otherwise do nothing. The syntax is therefore defined by the receiving object, not the sender.

All objects are arranged into classes. The concept of a class provides definition and grouping, acting as a static template description for storage of common data and method definitions for a group of objects. Every object is therefore an instance of a class. However it is possible to define classes that leave the implementation of certain methods undefined, these are termed abstract classes. Such a class can then provide a necessarily vague framework in which subclasses can refine the implementation to suit localised implementation. Such mechanisms provide for two important attributes of the object oriented paradigm: reusability and inheritance.

6.1.1.2 Reusability and Inheritance:

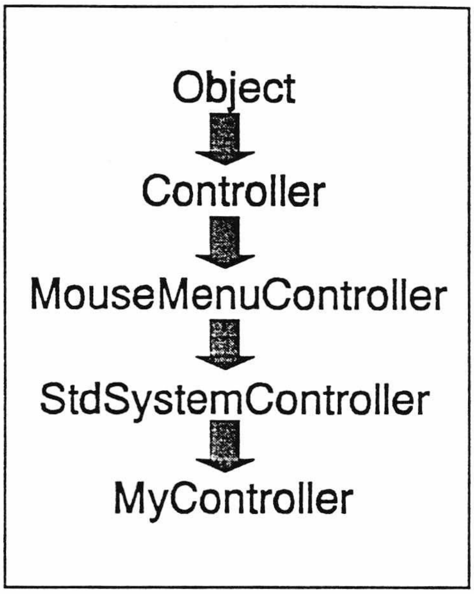


Figure 27 Inheritance Hierarchy for Class MyController

Classes are arranged in a tree structure with the root class providing functionality for all objects within the structure. Any object inherits the attributes/messages of its owner class but also of the parent classes within the class hierarchy. Similarly children objects of the current object inherit the parent object and class attributes. Functionality is provided by the local object being able to add to or override methods provided by default in the parent class, a technique referred to as subclassing. Any message received by an object is first checked against the current class for a match. If this is unsuccessful the messages inherited from the parent class are examined for a corresponding match, recursively to the top of the class hierarchy. If no match is found, the message is not understood by the system and a corresponding error produced. So in Figure Twenty Seven instance of *MyController* will respond to any of the methods available in *StdSystemController, MouseMenuController, Controller* or *Object* but still have the facility to subclass a method, for example the handling of menu options subclassed from the default implementation in *MouseMenuController*.

In developing an application, the aim is to look at the data first, not what is done *to* the data[211]. Objects and the corresponding messages that pass between them are identified to provide the functionality of the application. In programming terms, the aim therefore is to define classes and objects placed as high up the tree as possible. For example: A class Person may be defined in Figure Twenty Eight.

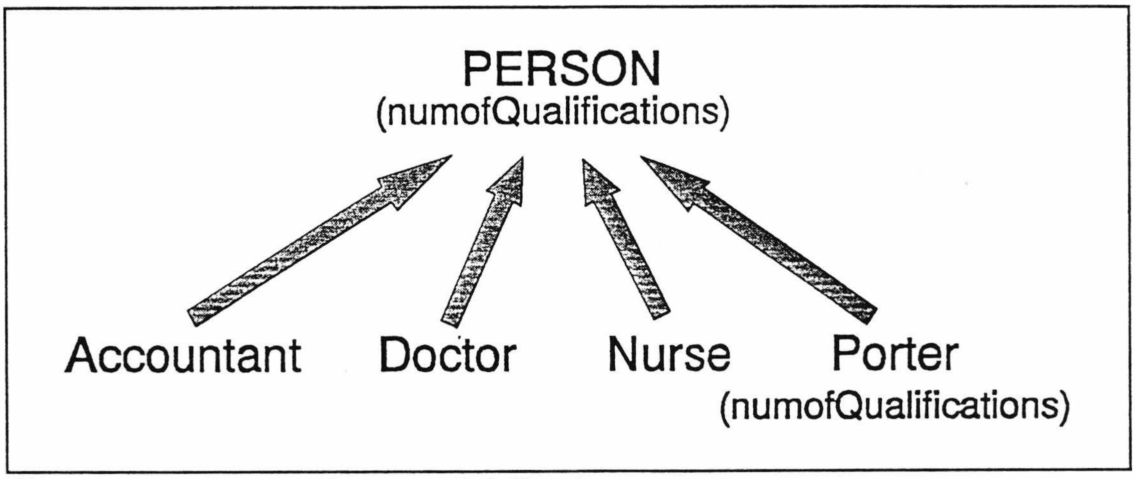


Figure 28 Example Message SubClassing

From the Person class these objects would understand the message "whoAmI" or "numOfQualifications". However to gain any functionality for the application, different instances have to react in individual ways. For example the application when responding to the numOfQualifications should be able to react in a different way if asking a Porter or an instance of Doctor. This can be implemented at the local class level, overriding the inherited message from the class Person. Within the class Porter, the corresponding method for numOfQualifications may be to respond with the number of years in the job. For the other objects, no new messages are required as they are inherited from the class Person by default.

6.1.1.3 Factoring and Polymorphism:

The object oriented paradigm provides a high level of code factoring via the inheritance mechanism. Factoring is the ability to provide code in a single location whilst maintaining reusability down through all subclasses and object instances. To maximise factoring, the programmer's aim is to place objects and methods as high in the class structure as possible, so that such classes and objects then become available to all sub classes and object instances via the inheritance mechanism.

In the functional decomposition paradigm, factoring is a byproduct of the decomposition technique. The notion is that by decomposing the implementation into smaller units, these units can be reused throughout the application. A limiting factor to the success of factorisation is the rigidity of the intermodule parameter passing mechanism, for example parameters have to be type formatted.

Polymorphism is used to denote where the same message can be broadcast to different objects in an application with each object responding in an individual manner, based on the local and inherited class structure. For the functional approach to programming, this rigidity in parameter passing restricts polymorphism to a very limited level, whereas in the object oriented paradigm, the interpretation of a message depends on the receiving object. For example, the message PRINT may be in the vocabulary of many objects: icons, windows and graphical objects may all respond to the message but in a range of different manners. When coupled with inheritance and factoring mechanisms, polymorphism can be described as high in the object oriented paradigm.

The key difference between these two paradigms is the focus of attention on what the data is, rather than on what should happen to it. The functional decomposition paradigm provides a high degree of functional abstraction with correspondingly little data abstraction. The object oriented paradigm provides data abstraction through object encapsulation and class structuring, and functional abstraction through message passing. Object orientation provides powerful facilities for code inheritance, encapsulation, message passing and polymorphism. Application development is focused around an incrementalist approach, building on class libraries of proven functionality rather than reinventing the wheel for each application, for example input/output or screen management routines. Application development in such an environment is therefore more closely linked to the real world problem, providing a flexible development method. For these reasons in particular, it has formed the software development platform for this project.

6.1.2 Object Orientation in a Graphical Environment

Data abstraction is a key provision in the object oriented paradigm. This is provided on the basis of data encapsulation and data hiding. For example a car driver need not know about the existence of the car engine, the only information/protocol that is required is "when pedal A is pressed the car travels forward in relation to the pressure applied". Thus all the information relating to the car engine is encapsulated within the engine object with the complexity and detail masked behind a strict interface protocol. This approach is adequate providing the user never needs to access directly a particular part of the engine, for example the gear box. This problem is highlighted if we provide a different user profile, for example a car mechanic. Here the car object could be described as a 'part-hierarchy', comprising sub objects, such as fuel, transmission and ignition system objects.

Direct manipulation is based on the provision that the structure of the application remains visible and directly accessible, whereas the object oriented paradigm requires an object to hide away its internal structure, presenting only a protocol to the outside application. There appears therefore, to be a conflict between the direct accessibility, for example with the direct manipulation style of interface design and the object orientation paradigm of data encapsulation and hiding[218]. Direct manipulation manipulates, in principle, the application structure providing direct access to any part, for example as a car mechanic or car driver. Each level of object within the part hierarchy must be accessible to maintain direct accessibility for different user profiles. If this is not so, any interface developed will be unable to exploit the direct manipulation environment to its full capability. Absolute information hiding within the part hierarchy structure is therefore incompatible with the direct manipulation style of interface design.

In this work a principle of data hiding is suggested based on various levels of data aggregation. Information hiding would then become contingent upon the user and task to be performed, for example maintenance on the ignition system of the engine by the mechanic or by the car driver. What information remains hidden and what is exposed via the functionality of the interface design would then be decided by the user, based on the task to be performed.

6.2 Choice Of Programming Environment

An initial project was developed on a PC IBM compatible to investigate the characteristics required for the final prototype environment. The development platform was Prospero Pascal[219] coupled to GEM (Graphics Environment Manager)[220] providing a standard library user interface toolkit (a Resource Construction Set)[221], for example slider and menu bars within a mouse driven environment. The project identified several limitations, not only of the hardware, for example speed, screen presentation and resolution, but particularly the underlying software development paradigm (and hence application structure). The development vehicle (Prospero Pascal) was a procedural language, providing a high level of procedural abstraction. Consequently this offered only limited scope for the development of user interaction design, with the environment heavily built around a single menubar main calling program.

GEM provided a pseudo direct manipulation environment, manipulating a screen rendering of the record structures hidden within the application. The screen images manipulated were created in addition to the internal data structures, adding to the complexity and size of the code design. The routines provided were of a low level (by comparison to object oriented approaches), for example screen management was explicitly required of the systems designer. Only very simple mechanisms of code reuse were provided, principally include files from the graphics library, resulting in very large volumes of code for even trivial applications. Such an environment deflected activity away from the focus of the project: user interface design to a complex information structure.

With the experience derived from this project a list of initial requirements of any development environment were drawn up. These included the following:

- Code reuse and incremental addition to a base of code already available to the application developer, particularly to avoid redeveloping the wheel each time, for example screen management tools

- High level routines coupled to the data on which they operate.

- Direct manipulation environment with full screen accessibility and pop up menu functionality

- Combined internal and graphical representation of data structures developed, minimising rendering problems

- Integrated development environment providing modular stepwise code development, that is not developed within an editor and then separately compiled and linked before execution

- Hardware consistent with a development of a large project and full screen direct manipulation, for example screen size and resolution

Two alternate development environments were available at the time, a combination of C source code programming coupled with the X Window Interface Development Library or application development in Smalltalk-80[222], a fully object oriented graphically driven environment.

Developing within the C environment provided the tried and tested path, with support, documentation and many source libraries available within the University. Resting upon a procedural language, the application would necessarily be structured within the functional decomposition paradigm, limiting the application design to a main calling program. The X toolkit[223] provides such a standardised interface protocol, based on a server/client model relationship. Several layers of code abstraction are provided from low level libraries of specific function operations to a high level 'widget' library of aggregated routines. Taking this approach, X does not provide any particular user interface style. Instead X supports many interface styles rather than a single interface look and feel[224], for example as compared to the GEM, Mac and Sun graphical libraries. A flexible set of 'primitive' level window operations provide device independence and functional reuse up through the layers of functional abstraction[225].

Smalltalk-80 provided an alternative approach to not only the development environment but also the underlying programming paradigm for the application itself. Smalltalk is a remarkably influential system that uses the object oriented paradigm most uniformly and completely[226]. Everything is an object, from an integer to a class description, with the concept of a main program replaced by objects and message passing.

Smalltalk provides more than just a language but a complete integrated development environment, providing an object oriented kernel, complete programming paradigm, and an object oriented user interface management system[227]. A complete library of high level tools and data structures is available for the systems developer to generate applications in a graphical integrated environment. A complete application toolkit is provided[228], for example an extensive series of graphical structures are already in place: overlapping windows, memory management and menu screen management. The philosophy of the application developer is to evolve additional incremental data structures harnessing what has been provided before, "standing on each others shoulders rather than each others toes". Many hundreds of classes and objects exist so that the task of the application programmer is to evolve additional data structures harnessing what has been provided before. Default interface handling mechanisms are provided which when coupled with the Model View Controller (MVC) paradigm provides for a powerful manipulative graphical interface environment.

The key problem with the Smalltalk approach is a direct result of its very appeal. Inheriting functionality necessarily requires a knowledge of location and structure of what is being inherited. The base Smalltalk-80 image provides approximately 68,000 objects that are all at the disposal of the application developer. A key problem in developing applications within such an environment where inheritance and subclassing form the central role to application development, is the very size of the image itself. This presents the application developer with a significant learning curve, besides mastering the underlying object oriented paradigm.

In summary, the top down paradigm is based on the functional decomposition of the application with the control vested in a 'main program'. The available development environments, particularly the C language coupled with the X Windows routine library provide a high degree of functional/procedural abstraction for the development of interface applications. Such environments are strongly vested in the structured programming paradigm, with each application requiring a 'main program', a memory manager, polling loops and a screen manager. Development work in such an environment requires the reinvention of these aspects for each application and artificial tailoring of the real world problem to the computer oriented architecture.

The object oriented paradigm follows a bottom up philosophy, based on what the data is rather than what happens to the data. The primary development environment is Smalltalk-80 which captures the object oriented paradigm most completely. Such a model follows the more 'natural' role of messages and objects in user real world actions. Smalltalk-80 provides powerful facilities for object inheritance, memory management, message control and polymorphism so that development work in such an environment is focused around an incrementalist approach, building on an extensive library of proven functionality. Application development in such an environment is therefore more closely linked to the real world problem, flexible in construction and based completely in an object driven environment. This project has therefore been developed entirely within the Smalltalk-80 environment, with the examples that follow taken from this environment.

More recently Interviews plus C++ have become available, coupling graphical functionality and window management to an extended C kernel[229]. The main base of C++ remains procedural in nature, for example one of the design guidelines was to have C code upwardly compatible with C++. Object oriented functionality has been made available as an extension, for example through function call overloading. This remains unable to provide a satisfactory alternative to Smalltalk-80, providing a coerced implementation of the object oriented paradigm[230] within a fragmented development environment (eg: linking, compiling editing etc). Smalltalk-80 remains the object oriented language to emulate, providing the cleanest, most complete implementation of the object oriented paradigm available[231].

6.3 Prototype Structure

Having finalised upon the Smalltalk-80 development environment, incorporating the object oriented paradigm for application design and development, this section will present the general design philosophy adopted and the final structure of the prototype developed.

6.3.1 Overall Design Philosophy

Smalltalk provides a unique integrated software development environment. Many development tools are provided within the graphical object oriented environment, for example browsers and inspectors, with stepwise interpretation rather than separate compile and execute approach to command discharge[232]. This project has attempted to exploit this platform, in particular the object oriented graphical environment to its fullest extent, incorporating particular object oriented data structures where possible, for example nontyped dictionaries.

As with any programming exercise this project evolved iteratively through several design stages:

- Identify the key objects within the application domain, developing an internal class representation
- Group these objects into placement location hierarchy within the existing class structure
- Identify the required functionality of each object class. Develop the methods required to depict this. Determine which of these require inheritance/subclassing and alter the class structure accordingly.
- Build dependency mechanisms as dynamic relationships between objects within the application.

One key challenge with Smalltalk programming is the required manipulation and interpretation of the size and complexity of the available class library. In developing any application the principle development method is through either the creation and attachment of new object class structures through this structure or the addition of methods to existing objects already located within the structure. New object classes can be added at any point in the structure, inheriting functionality from its parental hierarchy, whereas new methods provide new functionality directly to that object class as well as all child classes. Regarding software development, the latter approach of adding additional methods throughout the structure increases dramatically the complexity for the application developer in differentiation of the original image code from the additional application. An additional problem with application methods scattered throughout the structure is the backup and storage of the application during development, as well as on completion for the storage and porting to other Smalltalk-80 images.

The approach taken in this work has been to exclusively subclass structural development leaving the initial Smalltalk-80 image 'clean'. The newly developed classes are grouped into several categories related to the application, providing convenient development control and storage. The additional class structure provided is shown in Figure Twenty Nine.

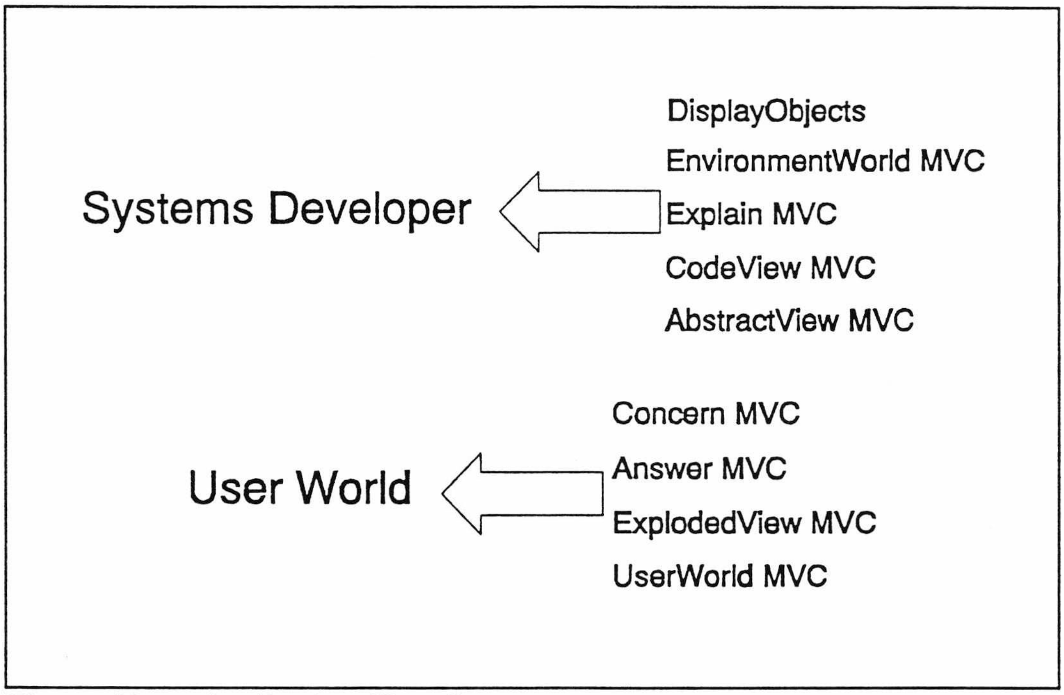


Figure 29 Application Class Structure

A second aim was to take advantage of the wealth of object oriented structures available in the base image. In procedural programming data structures are closely typed, providing uniformity and single pass compilation, for example an array of integer, file of type record. In an object oriented environment this restriction is removed, providing the application programmer with more scope to match the data structures to those of the real world. For example a collection may contain any multiple of any object, similarly a dictionary may reference any form of object. Many proponents of procedural languages suggest typed variables for program execution, providing more knowledge and hence control over the structure.[233] However the key issue here is the rigidity involved, for example a collection of objects should not be conceptually limited to a particular type, for example the contents of a container such as a drawer in a desk do not conform to a single 'type'.

6.3.2 Structure and Dependency Mechanisms

The key to providing the direct manipulation style of interface is the provision of a WYSIWYG environment[234]. This term however implies one side or half duplex communication between the user and the application. The target approach taken here has been similar to that of Took and WYSIWY...can get hold of and fiddle with. This implies not only the visual representation of what structure is available, but also the manipulation of its presentation and organisation of what it represents. To maximise this potential (thereby increasing the sense of user ownership felt by the user over the system) an aim in structuring the application has been to minimise off screen structures, focusing the structures around screen representation.

Screen presentation is achieved by the use of a *displayList* object that acts as a holder for objects to be displayed on the screen. A *displayList* provides the concept of screen location and updating functionality via the model/view/controller (MVC) mechanism.

Smalltalk provides a particular design strategy for the representation of objects, their display and control mechanisms. Such a three way factoring, termed the model view controller(MVC) paradigm, provides a strong modular approach to the user interface design function[235]. Three separate classes are provided for handling user interaction operations which when coupled with basic subclassing provide for screen object manipulation, for example pluggable windows.

Pluggability is a key form of object reusability/inheritance that is used in both MVC and menu structures[236]. For example, Figure Thirty shows the *SelectionInListView* structure, providing a complete view selection mechanism requiring only a menu string to be displayed and a string of corresponding messages to send to a designated model.

Three classes are required within Smalltalk to depict this modularity. A model class provides the specific central structure that has to be displayed within the view. Any object can serve as a model such that this may be as simple as an integer or as complex as a *displayList* acting as a holder for many other screen objects. The view classes handle the representation of the model upon the view. The model therefore has to be capable of answering basic requests of any view, for example printOn and storeOn. (default implementation is provided in Object for all objects). Views can be complex objects in themselves containing any number of subviews within a single superview, for example the superview *StdSystemView,* but primarily each view provides

SelectionInListView class
Methods for: 'instance creation'

on: anObject
printItems: flag1
oneItem: flag2
aspect: aspectMsg
change: changeMsg
list: listMsg
menu: menuMsg
initialSelectin: sel

Create a pluggable listView viewing anObject.

- printItems is set to establish form of message to send to anObject, if set to true the object is message os defaulted to printStings otherwise it is treated as a text like object already, ie: sending the messages #contents and #text.
- aspectMsg is sent to read the value of the current selection
- changeMsg is sent to inform anObject of a new selection
- listMsg is sent to read the current available list to be displayed
- menuMsg is sent to read the yellowButtonMenu associated with this view
- sel is sent to read the selection which is to be initially set as default.

Figure 30 ListInSelectionView method structure

protocol to establish how the model is to be displayed. It is the responsiblilty of the view to coordinate the MVC triad, holding private data that provides the bridge to the model and controller, for example the displaylist view holds reference to the model *displayList* and its controller through instance variables held in the superclass, *stdListController.*

A controller is required for each MVC triad to coordinate the user interaction with the model object via its view with a pointing device. The controller provides reference to the schedule of controllers (or controlManager) for co-ordination with all the other active controllers for views currently active on the screen, for example awaiting a mouse event. From the users perspective, the key use of the controller is the provision of mouse/menu interaction subclassed from the *mouseMenuuController* class. For example *SelectionInListController* provides standard pluggable instance variables for menu button operation with default inherited menu operations.

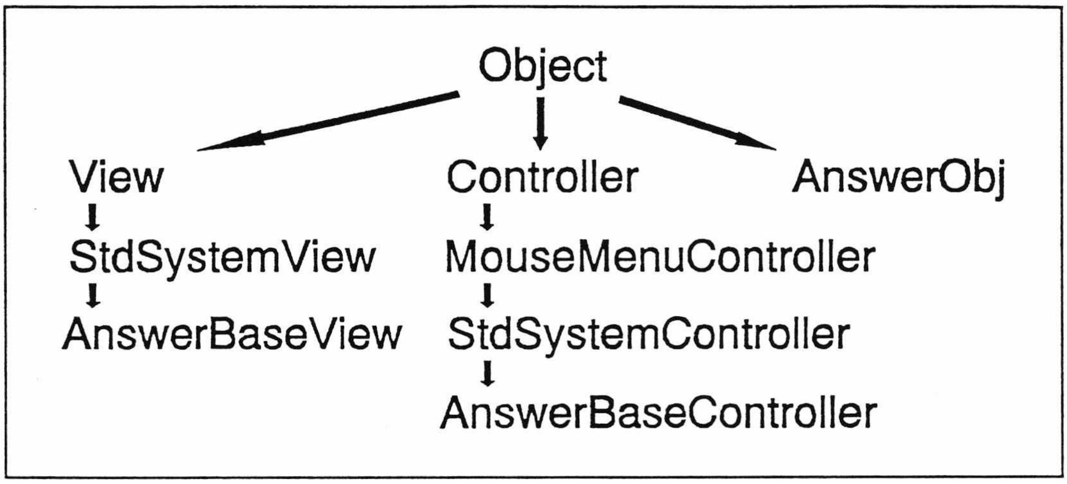


Figure 31 MVC Inheritance Hierarchy

A key component of Smalltalk that facilitates the MVC relationship is that of dependency[237]. Dependency provides a key mechanism of forming dynamic associations between objects within a structure that is in addition to explicit pointer based relationships, for example it forms a key component of screen object representation within the MVC framework. An example within this application is between an entity and its connecting link/relationship. A method of association was required by which a change in an entity, for example a movement about the screen, would generate a movement in any screen connected links. However not all changes in the screen representation of an object would require the screen links to update, for example to change a label. A combination of a dependency mechanism coupled with polymorphically defined updating of the dependent objects was used to provide this functionality.

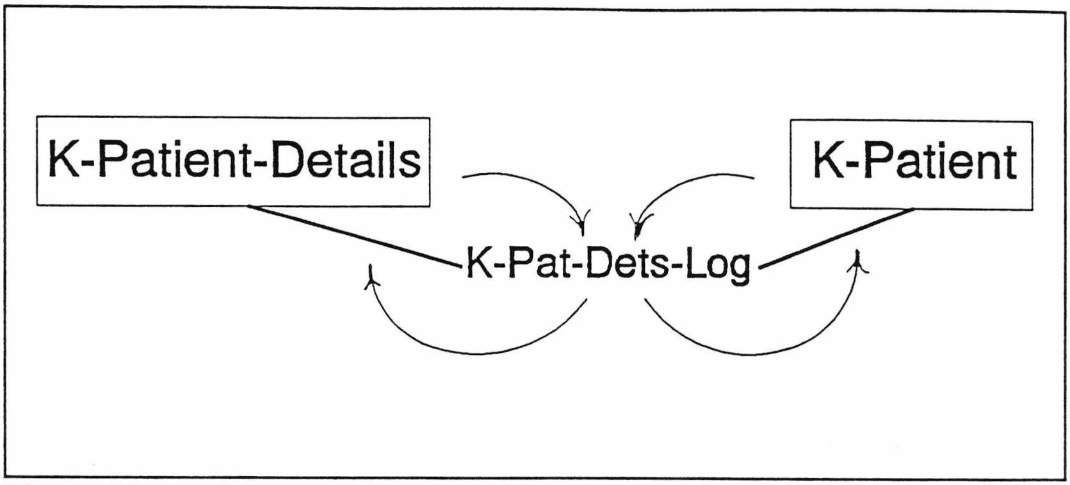


Figure 32 Link Dependency Explained

Here a link is dependent upon each of its ends so that the changed message is passed with a label indicating the type of change that has occurred. Each object in the dependency collection of the object, for example including the display list itself, receives the update message so that each may independently decide an interpretation and action required. The link would then ask the sender of the update for its new location, recalculating its end points for a redraw operation. If the update message indicates that for example the object has been renamed, then the dependent links may choose to ignore the message, whereas the displaylist would still be required to update.

6.3.3 Conclusion

Smalltalk maintains a rich coupling between the object oriented paradigm and the display device, providing a wealth of structures and methods for manipulating the user display device. The approach of this work has been to use these tools to their fullest extent, providing additional data structures not found in the functional decomposition environment.

6.4 Overall Data Structure of the Application

The classes provided within the application can be split into two categories: the basic core objects that are used throughout the application, for example entities, links and attributes that are primarily used to depict knowledge structures whether for the systems designer or the user; and object groupings that are primarily used to render the basic core objects onto the view, for example displayLists, areas of interest and concerns. Each principal object will be discussed here identifying inherited behaviour as well as key methods provided.

6.4.1 An Entity

This is a basic component for the representation of the semantic structure, providing a graphical representation of an entity within Chen's Entity-Relationship paradigm[182]. Spatial rendering of an entity is provided by inheritance of display characteristics from the *displayLabel* class and placement into a *displayList* structure. The entity class provides localised behaviour through subclassing, particularly concerning the attributes associated with an entity, for example the #Hide message to an entity requires local interpretation for the removal of the attributes displayed in addition to the entity itself.

However for example when an entity is moved through a view, any connecting linkages should be updated, preferably dynamically following the entity as it moves. Similarly if the entity is removed from the screen, any connecting linkages should be removed also. This is achieved by maintaining a dependency relationship between the entity and any linkages that are associated with it. When an entity changes in some manner all the connecting linkages are informed of the change reacting accordingly, usually by asking the entity for more information for example its new screen location.

6.4.2 An Attribute

Each entity provides the basic framework for the aggregation of associated attributes into a single conceptual unit. An attribute object provides the conceptual representation of an attribute in the Entity-Relationship Model. Such a graphical representation can be manipulated as any other screen object inheriting abstract behaviour from the *DisplayObject* class.

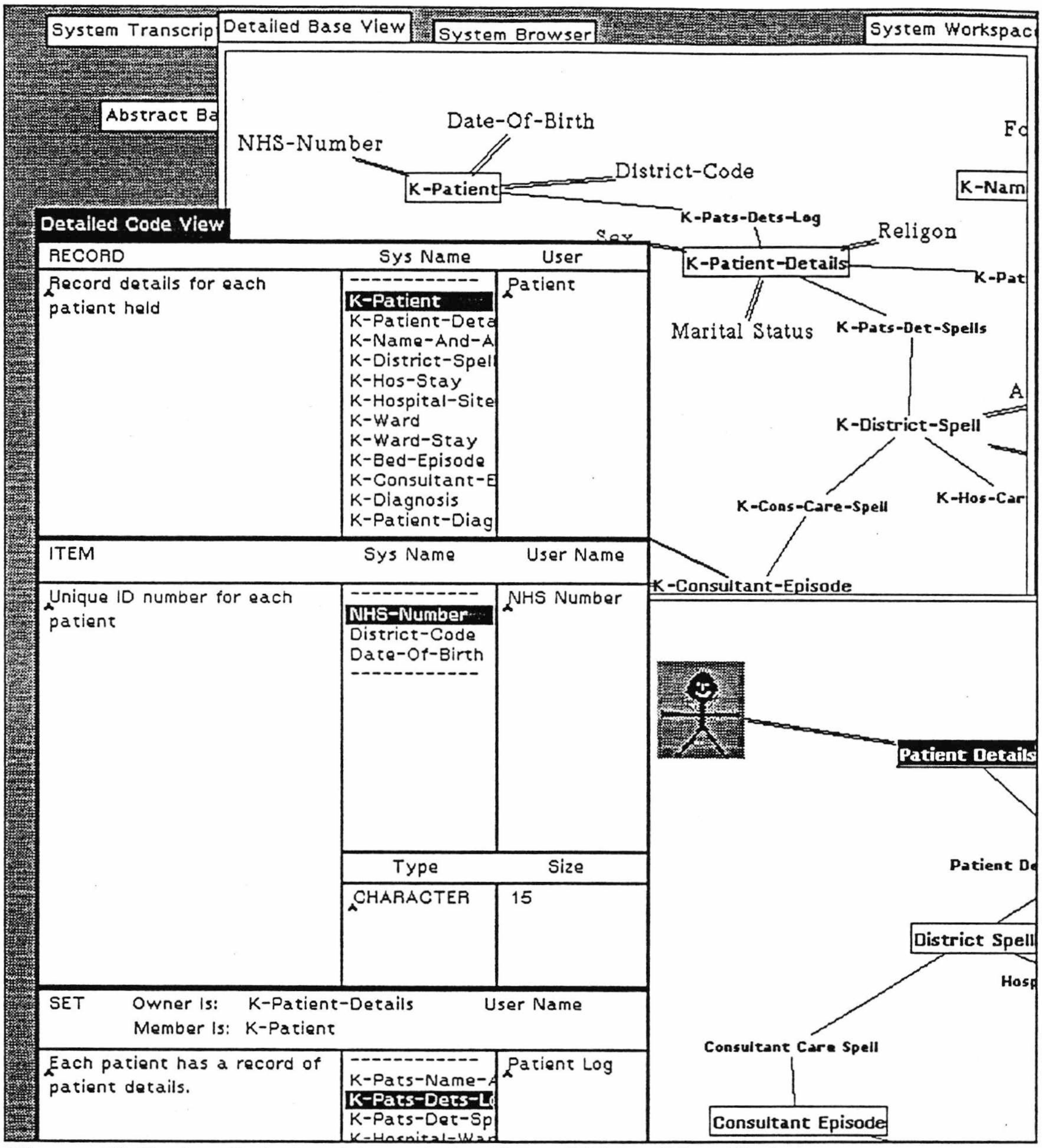
Subclassing provides for the addition of a selection condition against this attribute, this is held within an instance variable. An attribute is displayed connected to its entity by an 'attributeLink'. The associated entity has a direct reference to this attribute, for example opening and closing (exposing and hiding the attributes of an entity) providing direct updating for example when moving. The attributeLink is dependent on both the entity and the attribute receiving the update message.

6.4.3 Linkage

The remaining key component within the Entity-Relationship paradigm is the relationship. The link object is provided to depict relationships between any screen object. User defined linkages are available in which a macro path or series of mappings into the specific view are recorded. During query processing, these linkages provide a navigation through the specific view of the structure. For example, Figure Thirty Three shows K-Patient as an entity, NHS-Number as an attribute and K-Pats-Dets-Log as a linkage. The same information is displayed textually in the code browser alongside.

Each link holds a reference to both the attached entities, a link cannot have only one end. Regarding the systems designer side and data modelling expression, the first entity selected from which the link 'flows' may represent the singular form of relationship with the target entity representing the 'many' form of relationship. From Chapter Five it is assumed that this information is of little benefit to the user, effectively confusing the display. It has therefore been omitted from the user side completely. A similar approach has been adopted with the specification of the linkage nature (whether Fixed, Mandatory or Optional). Arguably this information provides little for the user at the expense of potential screen clutter and confusion. It has also been omitted.

A link object is composed of three parts, a main body or label that inherits from *displayLabel* and two linkLinkages that are subclassed *line* objects.



RECORD	Sys Name	User
Record details for each patient held	------------ K-Patient K-Patient-Deta K-Name-And-A K-District-Spell K-Hos-Stay K-Hospital-Site K-Ward K-Ward-Stay K-Bed-Episode K-Consultant-E K-Diagnosis K-Patient-Diag	Patient

ITEM	Sys Name	User Name
Unique ID number for each patient	------------ NHS-Number District-Code Date-Of-Birth ------------	NHS Number
	Type	Size
	CHARACTER	15

SET Owner Is: K-Patient-Details User Name
Member Is: K-Patient

Each patient has a record of patient details.	------------ K-Pats-Name-A K-Pats-Dets-L K-Pats-Det-Sp K-Hospital-War	Patient Log

Figure 33 Entity, Relationship and Attribute Objects Identified

Subclassing provides for a local interpretation of this structure, for example the move-update dependency requires the linkage body to move, generating two redraw messages for the linkLinkage. These redraw messages are subclassed to provide a local implementation of redrawing endpoints. An additional presentation problem related to this issue has been the determination of the endpoint of a link as its connected entity moves. Assuming the end point of an entity is its origin, this would provide for poor presentation with links crossing the owner label to reach the origin. The implemented solution allows the link to trace around the label border as either the label itself moves or the other end of the linkage.

6.4.4 Description Object

This object provides the ability for textual input onto the semantic model tying into any object using a link pointer. Any form of textual information can be entered, from the description of what an object represents to memory joggers or notes about particular courses of action available at that point. In the future, other forms of object could be incorporated here, not only text but voice or pictorial objects, building a wider complete picture of the concern at hand. Subclassing from *stringHolderView* and *StringHolderController,* this object provides a wrap around text block fitted into a user specified area on the map.

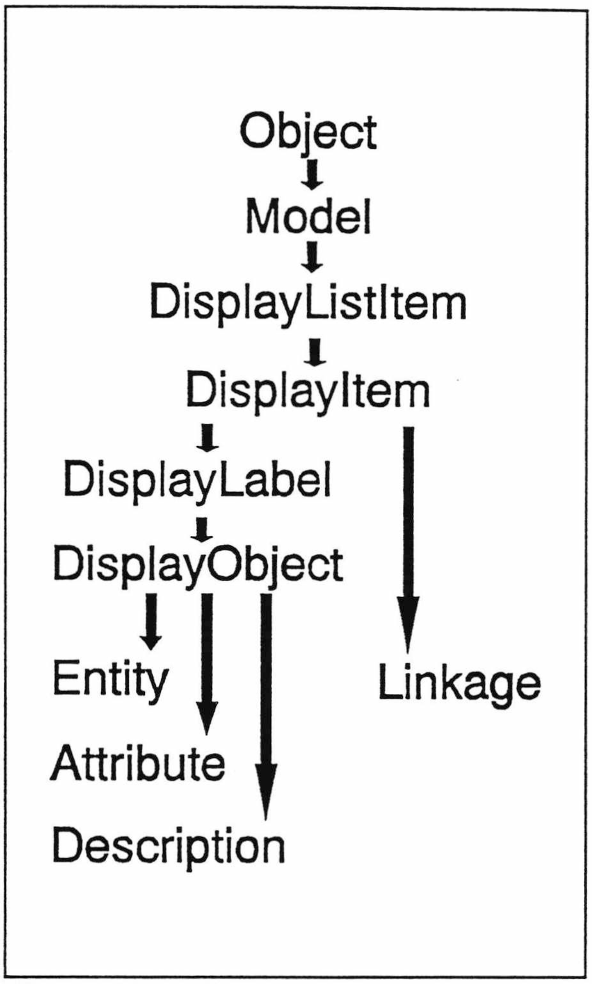


Figure 34 Class Hierarchy For Core Objects

The objects described so far are all the basic core objects that are used throughout the application. Each is a form of screen object, inheriting functionality from the *DisplayObject* class. The class structure is shown in Figure Thirty Four.

6.4.5 Area Of Interest:

An area of interest is a screen object whose main task is to provide a mapping between the general and specific conceptual models within the systems side of the application. It provides a representation within the generalised view of a captured group of entities from the specific view. Such groupings are initially defined by the systems designer but in the final end user side of the application these are available for refining or as a basis for the creation of additional areas of interest which allow a close tie to personal and task based interpretation. The principle is that such a grouping should be comprised of objects which 'naturally' fall or occur together within the users application domain.

The abstract object class provides a visual representation of the area of interest within the abstract conceptual view. For example Figure Thirty Five indicates the grouping of entities (boxed) that make up the abstract object patient details (highlighted).

Subclassed from the *DisplayObject* class this object provides all the basic screen movement operations with refined methods for screen movement and copying, principally to handle the grouping from the specific model.

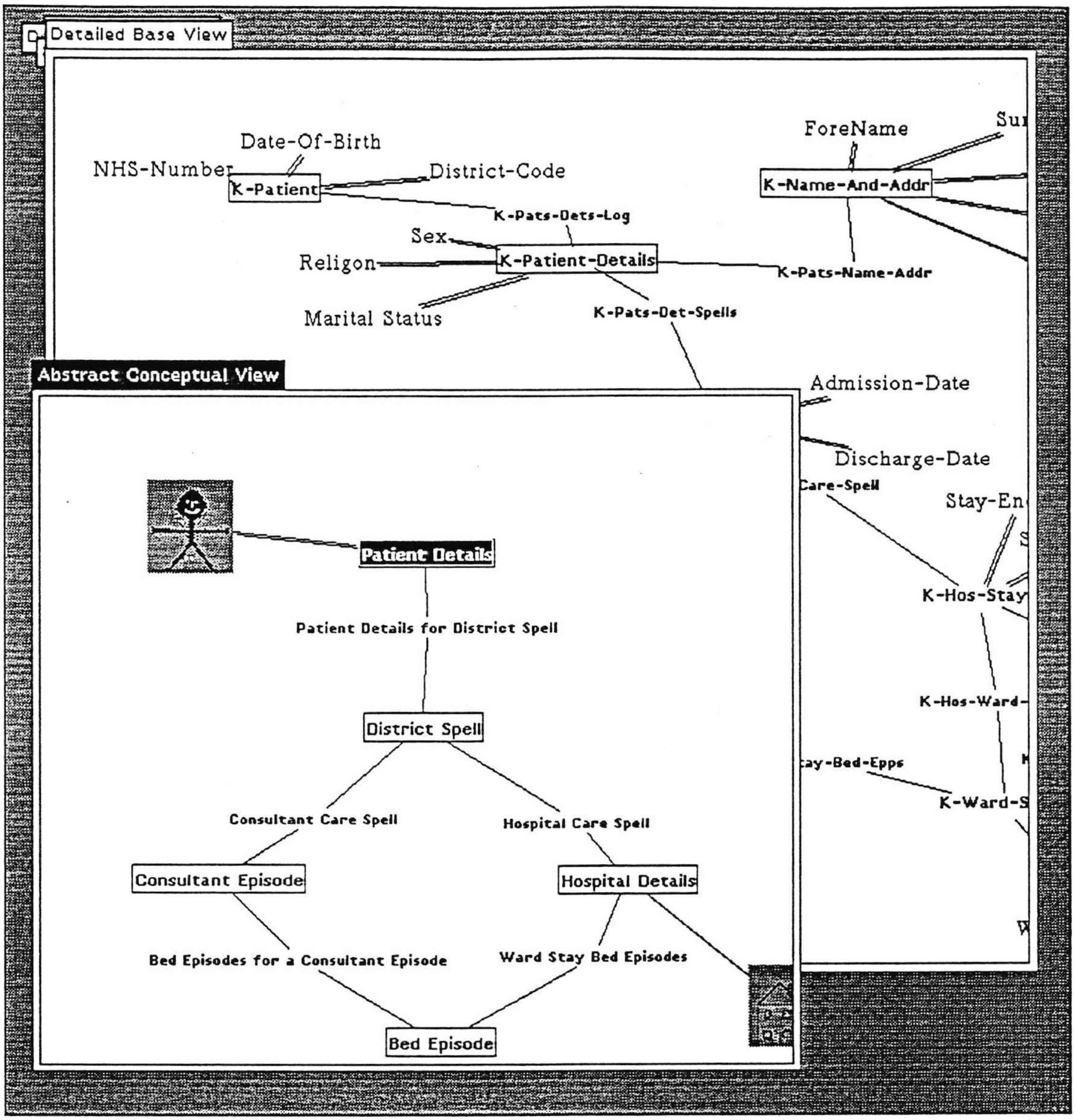


Figure 35 Relationship Between Area of Interest and Specific Conceptual View Defined.

The grouping of the specific objects is encapsulated within the abstract object as an instance variable. Any specific level object may appear in any number of abstract entities, that is: have any number of parent abstract objects. Such a provision mirrors the real life situation of objects occurring in different interest domains, for which special provision has to be made when opening up a concern. (See Chapter Five)

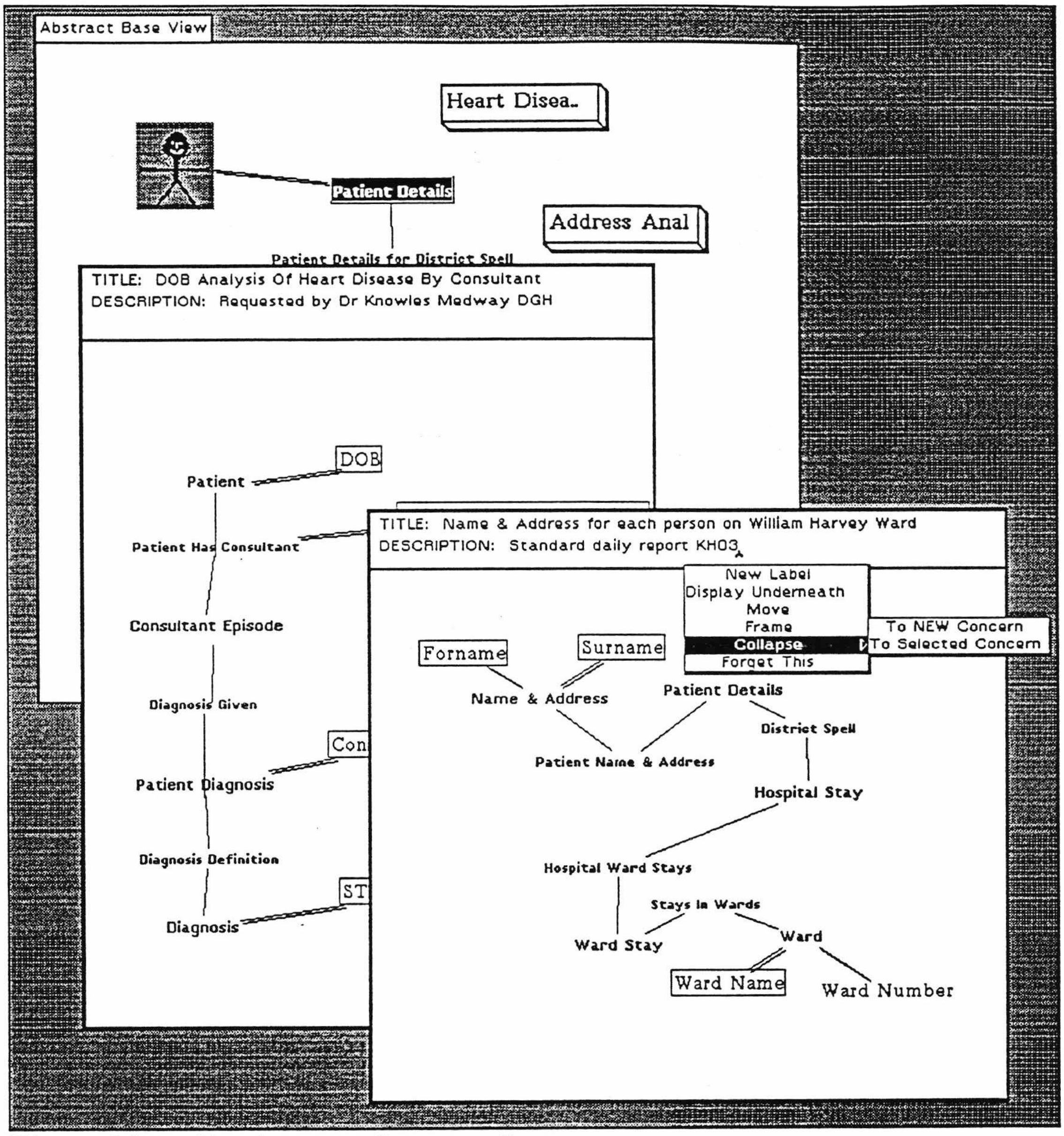
6.4.6 A Concern

This is the primary end user tool enabling the aggregation together of information objects, abstracting away from the base structure into a constrained problem domain. In this way a concern provides the aggregation of information into a spatial environment enabling the user to concentrate upon a small and confined subset of the complete application structure. This is reinforced through the physical separation of the chosen areas of interest into a new, separate view with its own menu functionality.

Once opened the user can manipulate this graphical structure, for example moving, hiding, renaming, adding linkages and descriptions with the aim of tailoring the representation towards a combination of personal and task based interpretation.

Any number of concerns may be open at a time and one concern can be revised from, or added to another. For example if there is already a concern covering part of the request the user may wish to aggregate that concern directly with additional areas of interest or with other previously identified concerns, to generate the required graphical presentation. A logical extension, which has not been implemented, would be to allow the user to copy, cut and paste between concerns.

When collapsed, the concern MVC (via the view object that holds reference to its model and controller) is loaded into an instance variable within a button



TITLE: DOB Analysis Of Heart Disease By Consultant
DESCRIPTION: Requested by Dr Knowles Medway DGH

TITLE: Name & Address for each person on William Harvey Ward
DESCRIPTION: Standard daily report KH03

Figure 36 Simultaneous Concerns Open

object and added to the original base list. This enables the information within the concern to be capitalised into the abstract base list for future use aiding overall interpretation of the information held within the application. Simultaneously a degree of separated abstraction is maintained.

6.4.7 Mother Object:

The mother object provides coordination between the abstract and base views within the system designers side of the application. Particularly this object handles the generation of areas of interest, capturing the mapping between the abstract and specific conceptual views from the systems designer. It provides the internal link between an individual area of interest within the generalised view and a group of specific entities in the specific view handling screen presentation. For example selection of the abstract area of interest requires the corresponding group of specific entities within the specific view to be boxed. The mother object gives manipulation of a buffer object enabling specific entities to be created without an immediate direct reference to an area of interest. Subsequently the mother object provides for these buffered objects to be grafted onto or between areas of interest.

6.4.8 An InBetween Object

This object provides the internal link between the base list and the code view in this system side of the application. The code view and the base list provide two views of the same model - the *displayList*. However the model contains several different object instances whereas the code view is only required to handle the textual description of entities, attributes and their relationships. An object to manage this relationship is required, termed an inbetween object. This object manages interaction of the textual and graphical displays, for example when creating an object via the textual display the graphical display has to be updated and vice versa.

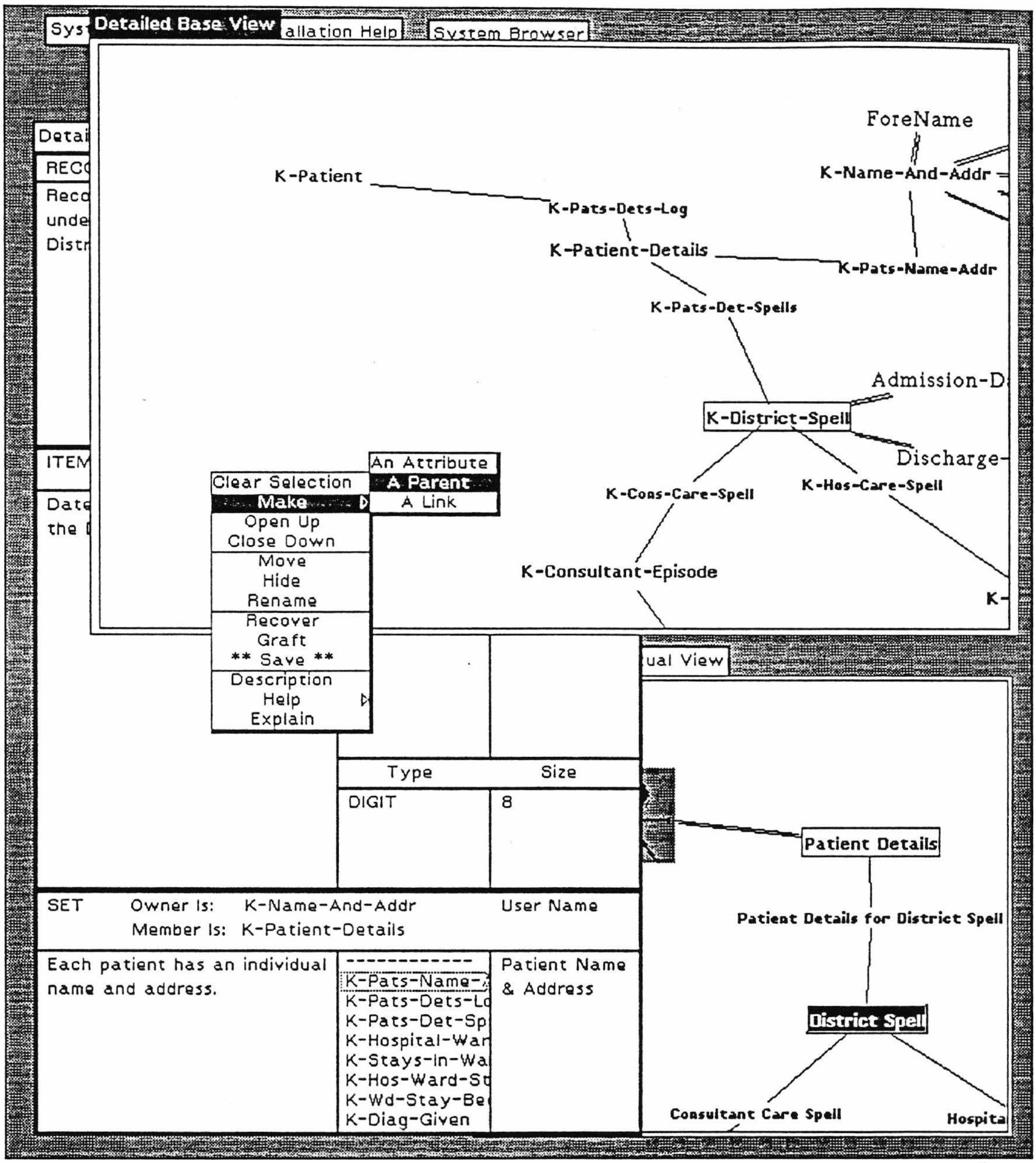
6.5 Systems Developers Side of the Application

6.5.1 Overall Description

Three views are provided with the principal aim of capturing the mapping between the abstract and the specific conceptual views providing data description language (DDL) code as a byproduct. The views provided follow this arrangement closely, supplying spatial representation of the abstract and specific (or base) conceptual views with a code view for textual browsing of the specific data structure.

The main user[a] interaction is to be with the abstract and base views, building a graphical representation of each corresponding model for the application. This can be approached in several ways: by initially building the base view and mapping this information into the abstract view as a secondary process (bottom up), by taking a more top down approach identifying the abstract objects within the application and mapping these into the base view, or more likely a combination of these two approaches. During this process there is large scope for error checking of semantic and logical correctness of the structures being developed. The prototype provides only simple graphical checking, for example ensuring that all entities are attached within the base list, (unattached entities are simply discarded) and for missing linkages between abstract entities. It is therefore possible for the data modeller to develop an abstract conceptual view without any linkages at all, relying on the

[a] User in this section refers to the systems developer



Type	Size
DIGIT	8

SET Owner Is: K-Name-And-Addr User Name
Member Is: K-Patient-Details

Each patient has an individual name and address.

K-Pats-Name-
K-Pats-Dets-Lo
K-Pats-Det-Sp
K-Hospital-War
K-Stays-In-Wa
K-Hos-Ward-St
K-Wd-Stay-Be
K-Diag-Given

Patient Name & Address

Figure 37 System Developers Overall Presentation

system to fill them in as a secondary process. Figure Thirty Seven represents the typical layout presented for the systems developer who is in the process of specifying part of the ICL DIS database (as presented in Chapter Three). All the views inherit standard Smalltalk window functionality from the MVC triad classes *standardSystemController* and *standardSystemView*. The inBetween class provides the relationship between base view interaction and the respective browsers in the code view. Similarly the mother class provides the relationship between the abstract and base views, particularly regarding the manipulation of areas of interest.

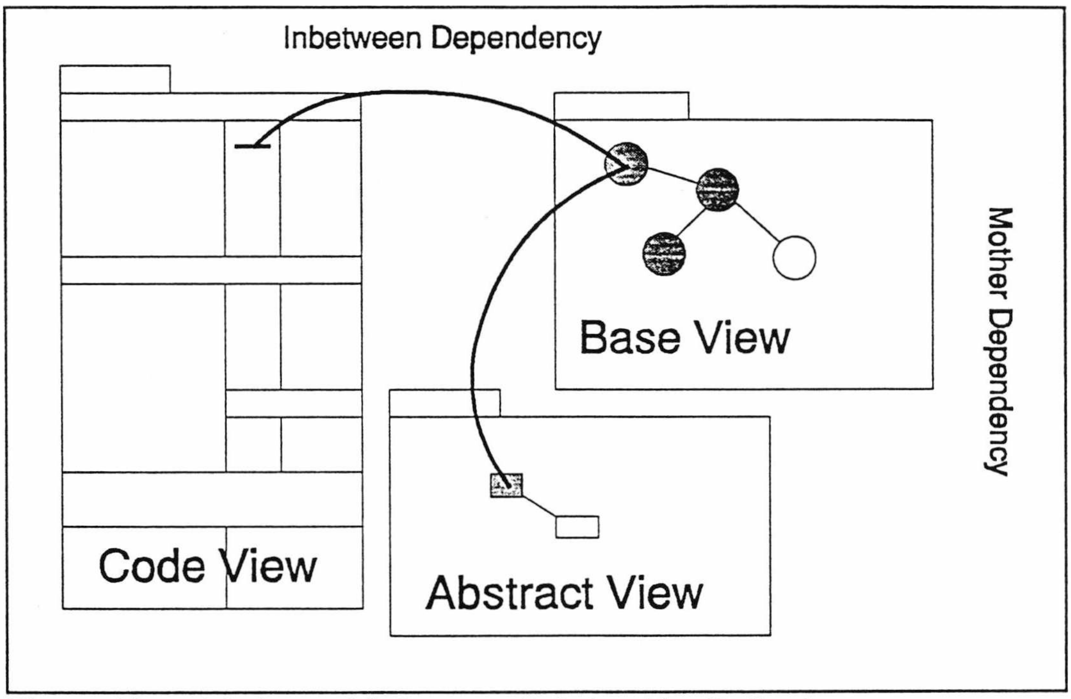


Figure 38 Mother and InBetween Relationships Defined

6.5.2 Abstract User View: Description and Aims

The abstract view provides representation of the abstract conceptual view derived from the ANSI/SPARC data modelling process[181]. This will form the base start up view for the end user environment which the user may tailor or add to, providing an interpretation of the information structure. In its completed state, this view is assumed to be the base interpretation of the information structure from the system modeller's perspective, for example all labels are in system terms rather than final end user terms.

6.5.2.1 Internal Structure

This view is subclassed from the superclass pair *standardSytemController* and *standardSystemView,* providing inherited window management via a bluebutton menu, for example window movement, sizing and framing with pluggable yellow and red button menu structures. The model is the *displayList,* an *orderedCollection* holding the various screen objects to be displayed.

6.5.2.2 Menu Functionality

Three mouse buttons are provided, the protocol for each being inherited from the *mouseMenuController* structure. The red(left) button following Smalltalk convention is provided as a point and select device. In this view only one object can be selected at a time which is then stored in the view object itself. If another object is selected (or no object at all) the current selection is deselected, that is inverted and the new object inverted in its place. Such an inversion provides an update broadcast to any dependencies from which the Mother object updates the boxed selections in the base list. The yellow (middle) and blue (right) menu buttons each provide a pop up menu on the display. The functionality of each is defined below.

<u>Yellow Button (Pluggable - subclassed locally)</u>

Make Abstract Entity — Generate an abstract entity object that will represent an area of interest in the specific view.

Link — Generate a linkage from this selected object to the next object that is clicked over.

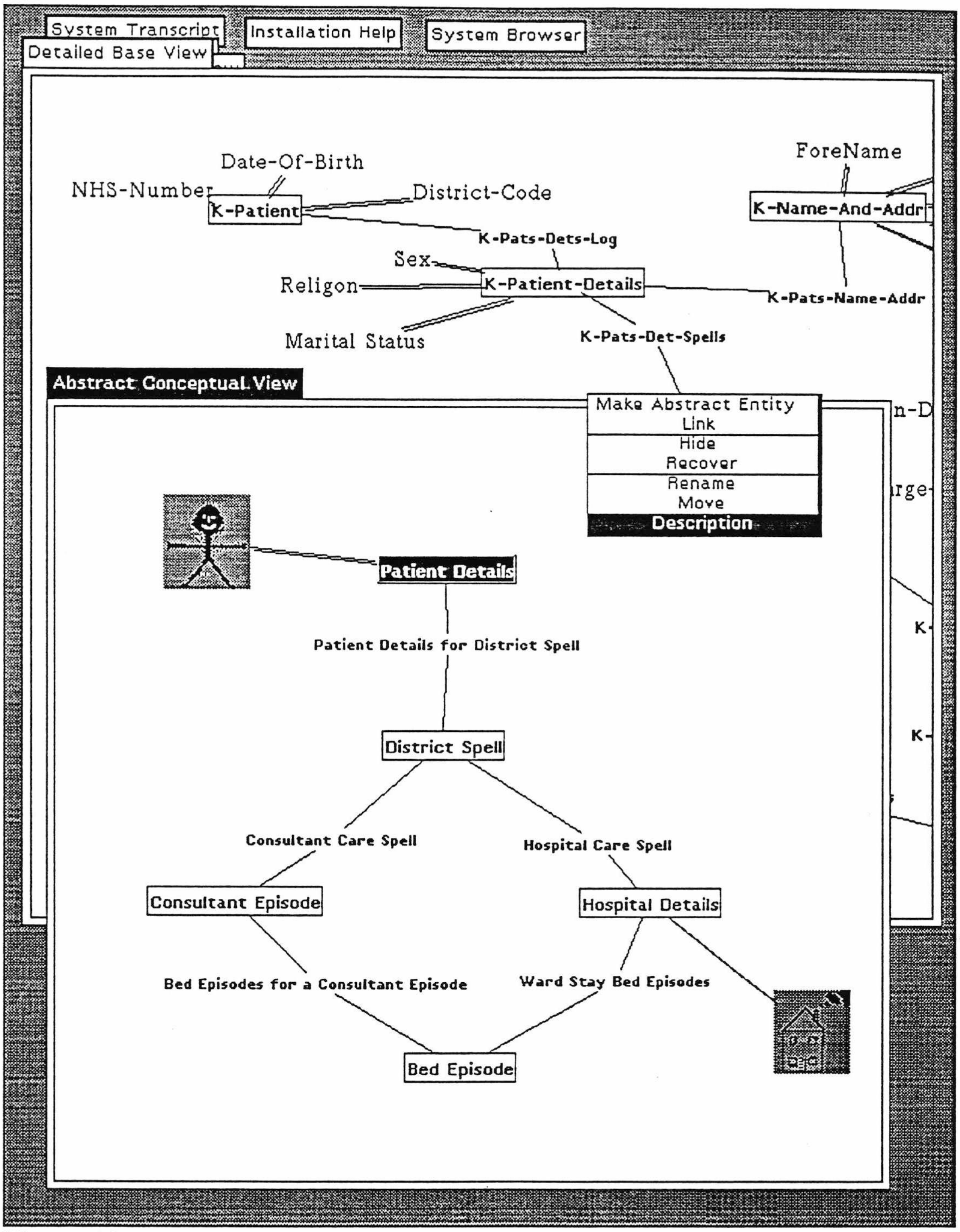


Figure 39 Abstract View Presentation

Hide — Remove the selected objects from the *displayList* to a buffer for possible recovery.

Recover — Recover the last object from the buffer and redisplay back on the view

Rename	Give a new name to the selected screen objects.
Move	Move the selected objects together at the cursor dragging them until any mouse button click.
Description	Build a description object

Blue Button (Inherited from MouseMenuController)

New Label	Provide a new label for this view.
Under	Push this view to the bottom of the stack of views on the display.
Move	Drag the view to another location on the screen maintaining proportions.
Frame	Alter the size and shape of the view.
Collapse	Collapse the view to an icon based on the given view label.
Close	Remove view and contents from the screen.

6.5.3 Base View - Description and Aims

The base view gives a graphical representation of the specific conceptual model. This view will form the basic structure into which all user abstractions and representations will be mapped, for example concerns and areas of interest.

6.5.3.1 Internal Structure

Like the abstract view, this view is subclassed from the superclass pair *standardSytemController* and *standardSystemView* providing inherited window management via a blue button menu, for example window movement, sizing and framing with pluggable yellow and red button menu structures. The model is the *displayList*, an *orderedCollection* holding the various screen objects to be displayed.

6.5.3.2 Menu Functionality

As with the Abstract view, the menu functionality is provided over the three mouse buttons: red(left) follows Smalltalk convention operating as a selection device, yellow(centre) and blue(right) provide pop up menus as follows.

<u>Yellow Button (Pluggable - subclassed locally)</u>

Clear Selection	Clear the selection.
Make -> An Attribute	Create an attribute at the mouse click position.
-> A Link	Create a link starting at the last selected object finishing on object with next mouse click.
-> A Parent	Create an entity at the mouse click position.
Open Up	Display the attributes relating to an entity.
Close Down	Remove from the display the attributes associated with the selected entity.
Move	Move the selected objects together at the cursor dragging them until any mouse button is clicked.

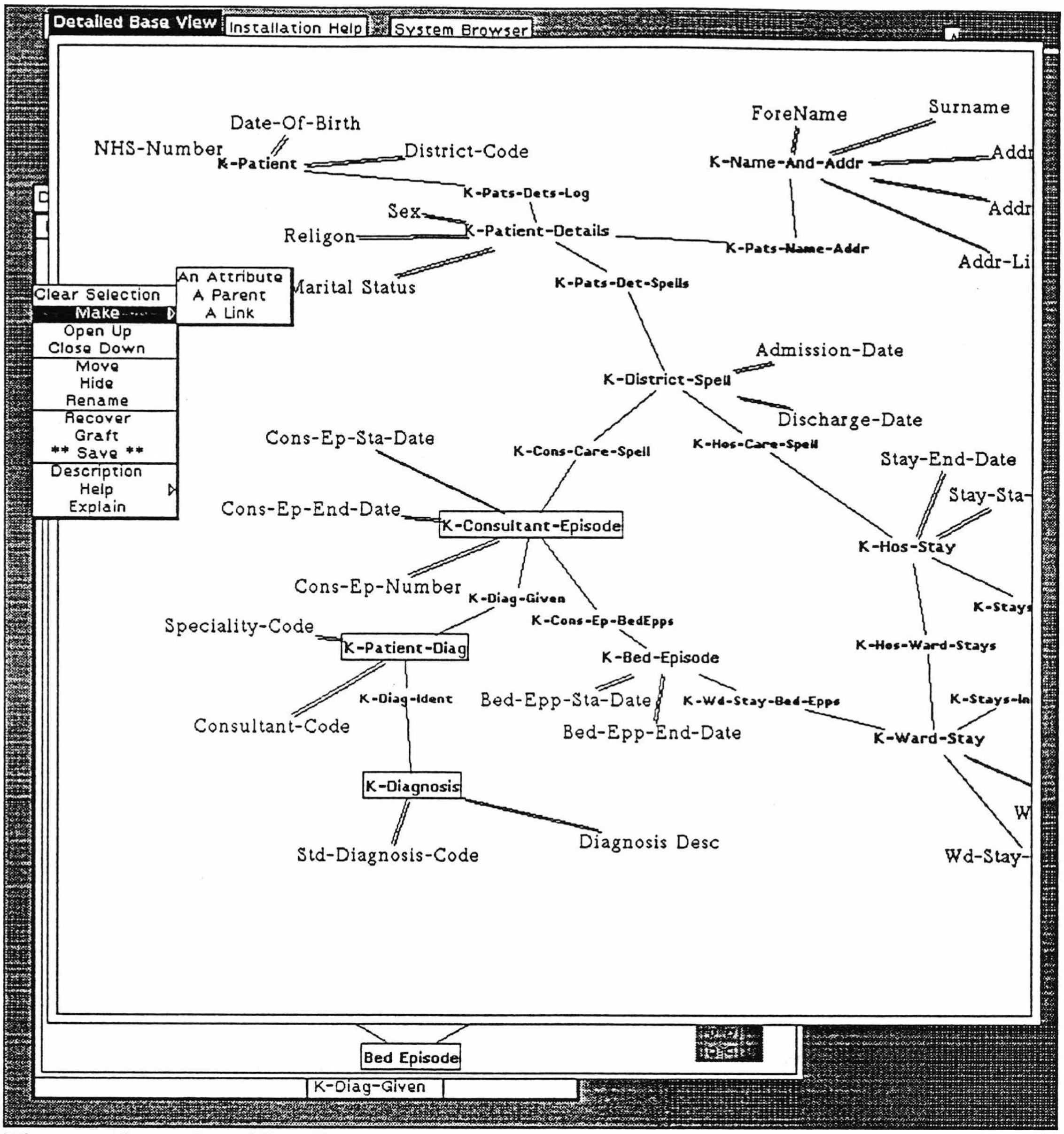


Figure 40 Specific View Representation

Hide
Remove the selected objects from the *displayList* to a buffer for possible recovery.

Recover
Recover the last object from the buffer and redisplay back on the view.

Rename
Give a new name to the selected screen objects.

Graft
Add the selection in base list to the selected area of interest definition. If no area of interest is selected then graft the selection in the base list onto the buffer removing all other association.

** Save **	Save the views generated to disk. Output DDL code to a separate file.
Description	Build a description object.
Help -> Instructions	Provide a list of direct available instructions from here
-> Explain	Provide a general descriptive help for this point.

Blue Button (Inherited from MouseMenuController)

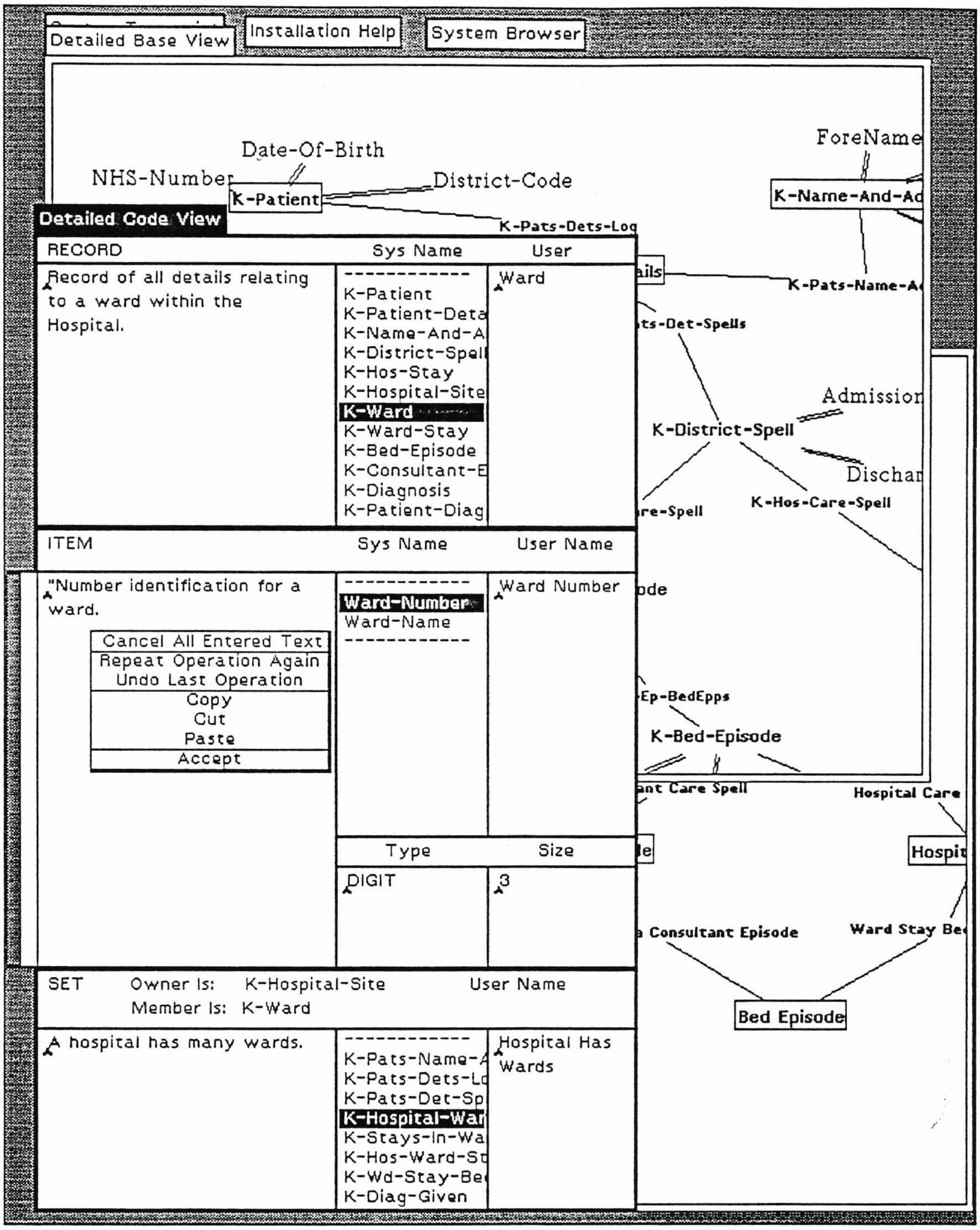
(see Section 6.5.2.2)

6.5.4 Detailed Code View: Description and Aims

This view provides a textual browser environment over the data description code that is generated as a byproduct from the user activities within the specific view. The code view is a complex arrangement of subviews (MVC's) arranged into three groups: RECORD, SET and ATTRIBUTE.

The RECORD view provides a text browser over the entities within the specific view. Entities are selected via system or user name within a *selectionInListView* MVC. The description view which is dependent on this selection is then updated. Each entity holds a description instance variable that provides the textual input for the Help/Explain function.

Each view has an individual controller menu allowing update of any displayed item. An option within the main *selectionInListView* browser enables the user



RECORD — Sys Name — User

Record of all details relating to a ward within the Hospital.

K-Patient
K-Patient-Deta
K-Name-And-A
K-District-Spell
K-Hos-Stay
K-Hospital-Site
K-Ward
K-Ward-Stay
K-Bed-Episode
K-Consultant-E
K-Diagnosis
K-Patient-Diag

Ward

ITEM — Sys Name — User Name

"Number identification for a ward.

Ward-Number
Ward-Name

Ward Number

Type: DIGIT

Size: 3

SET Owner Is: K-Hospital-Site — User Name
Member Is: K-Ward

A hospital has many wards.

K-Pats-Name-A
K-Pats-Dets-Lo
K-Pats-Det-Sp
K-Hospital-War
K-Stays-In-Wa
K-Hos-Ward-St
K-Wd-Stay-Be
K-Diag-Given

Hospital Has Wards

Figure 41 Code Browser View

to create an entity from within the DDL environment from which the system derives a default representation on the specific view.

The ATTRIBUTE browser is similar in construction and operation to the RECORD browser above, except that the main *selectionInList* model is dependent upon the selection made in the RECORD browser. Only when an entity is selected will the associated attribute list become available as a model to the main *selectioninlist* for this browser. Only when this occurs will the other dependent views be updated. Again each of these views takes an instance variable as its model, for example attribute type, size or description.

The SET browser provides a browsing function over the relationship links held within the specific view. The construction and operation is very similar to that of the RECORD and ATTRIBUTE browsers described above, but an additional view is provided, indicating the owner and member of the linkage.

6.5.4.1 Internal Structure:

The code browser is an aggregation of three similar browser structures organised into a *standardSystemView* and controller pair providing standard blue button window management, for example closing, moving etc.

Each browser is composed of five model/view/controller triads, each with its own pluggable menu structure. Each browser is essentially the same so only one will be analyised for internal structure here - the ATRIBUTE browser. Each area of the view requires a MVC triad to subclass yellow button functionality.

The seven MVC triads required are as follows:

- There are two title views, each is a string of text with no controller - no yellow menu button is available

- The main *selectionInlistview* is itself two MVC combinations requiring two lists from the overall browser model, a list for selection and a second list for target answer. Each is derived by interrogating the overall model that replies with two ordered collections, a collection of system names and a corresponding list of user names from which the list browser is constructed.

- Three stringHolder MVC's each of which take an instance variable from the browser model as their model which is then directly updated.

6.5.4.2 Menu Functionality

Three mouse buttons are provided, the protocol for each being inherited from the *mouseMenuController* structure. The red(left) button following Smalltalk convention is provided as a point and select device depending on the view. For example, within the text view the red button acts as a pointing device, locating the text cursor, whereas in other views, for example the *selectionInListView,* the red button acts as a selection device. The blue button operation is inherited from *mouseMenuController* with the yellow button operation subclassed providing a pluggable view for each MVC triad. Only the RECORD browser is described overleaf:

RECORD

Description View

Cancel All Entered Text	Remove all typed in this view so far.
Repeat Operation Again	Repeat last operation, eg: replacement
Undo Last Operation	Undo the last menu operation performed
Copy	Copy highlighted text to buffer
Cut	Remove highlighted text to buffer
Paste	Empty contents of buffer to cursor location
Accept	Store entered contents to model

SysNameView

Add Record	Generate an entity
Paste Back Record	Reinstate removed buffer contents
Rename Record	Rename the entity selected
Remove Record	Remove an entity to the buffer

UserNameView

(as DescriptionView above)

6.5.5 Summary

This approach has provided a direct manipulation environment whose aim is to capture in a graphical form, the mapping between the generalised and the specific conceptual models during the data modelling process. The DDL code required for database construction is provided as a byproduct within a fully browser driven environment for fine scale tuning. The modeller is given scope to work from the abstract to the specific, from the specific to the abstract, from the code level to the specific or as is more likely a combination of these.

From this process the abstract view will provide the initial base interpretation of the information held within the application structure. The specific model developed will provide the lowest level interpretation into which all the user aggregation mechanisms, areas of interest and concerns will be mapped and subsequently translated into database specific query code.

6.6 User Side of the Application

In this section the structure of the application relating to the end user is presented by means of building on the health authority DIS data model example provided in the previous section. Three main components are introduced, the base view, a concern view and the query processing system.

6.6.1 Overall Description

Two views form the main component of this section. The first is the base view, the generalised conceptual model representation from the systems designers specification. The system modellers interpretation provides a basic start up interpretation of the information in the structure, similar to an organisation chart. An organisation chart is fixed in time, based on a static interpretation by another person of what in theory is available. This is the principal reason for this view acting only as an initial contact with the system, a base that is added to as the user progresses building concerns and issuing queries just as an organisation chart is developed from the formal to the informal, through experience.

The user is free to update this base view in any way, either through directly manipulating the base display or through the development of queries and concerns (described in more detail in sections 6.6.4). This degree of freedom is provided within a controlled access structure based on user access to only those areas of interest displayed. Areas of interest in which the user is not allowed access are simply not displayed.

6.6.2 Base View

6.6.2.1 Description and Aims

This view provides an overall graphical presentation of the structure of information within the application. The contents of this view originate from the systems side of the application, the abstract view based on the generalised conceptual model. The origin of the view acts however only as an initial starting point for the users interpretation of the information structure held within the application. As queries are expressed and the screen presentation altered, this view will develop into a personalised interpretation of the information system within the application.

To enable this, the user is provided will full screen control over all the screen objects provided, for example move, hide, rename as well as scope to create new screen objects, for example descriptions and linkages. These tools provide for the tailoring of the view towards an individual job function, for example presentation of areas of interest may be based upon access permissions as well as providing scope for tailoring the view towards the user's own interpretation, for example recombining the areas of interest into additional areas of interest (see section 6.6.3), altering object representation or expressing information linkages between screen objects from a user, rather than a system perspective.

6.6.2.2 Internal Structure

As with all the base views within the application, this view inherits directly from the *standardSystemController* and *standardSystemView* MVC pair. The yellow button menu is subclassed (plugged) for this structure providing localised menu functionality. The model for this view is a *displayList*, copied from the systems side of the application via disk storage with dependency stripped. This representation is provided as an initial startup view only, for example that provided in Figure Forty Two is taken from Figure Thirty Nine.

6.6.2.3 Menu Structure

Three mouse buttons are provided, the protocol for each being inherited from the *mouseMenuController* structure. The red(left) button following Smalltalk

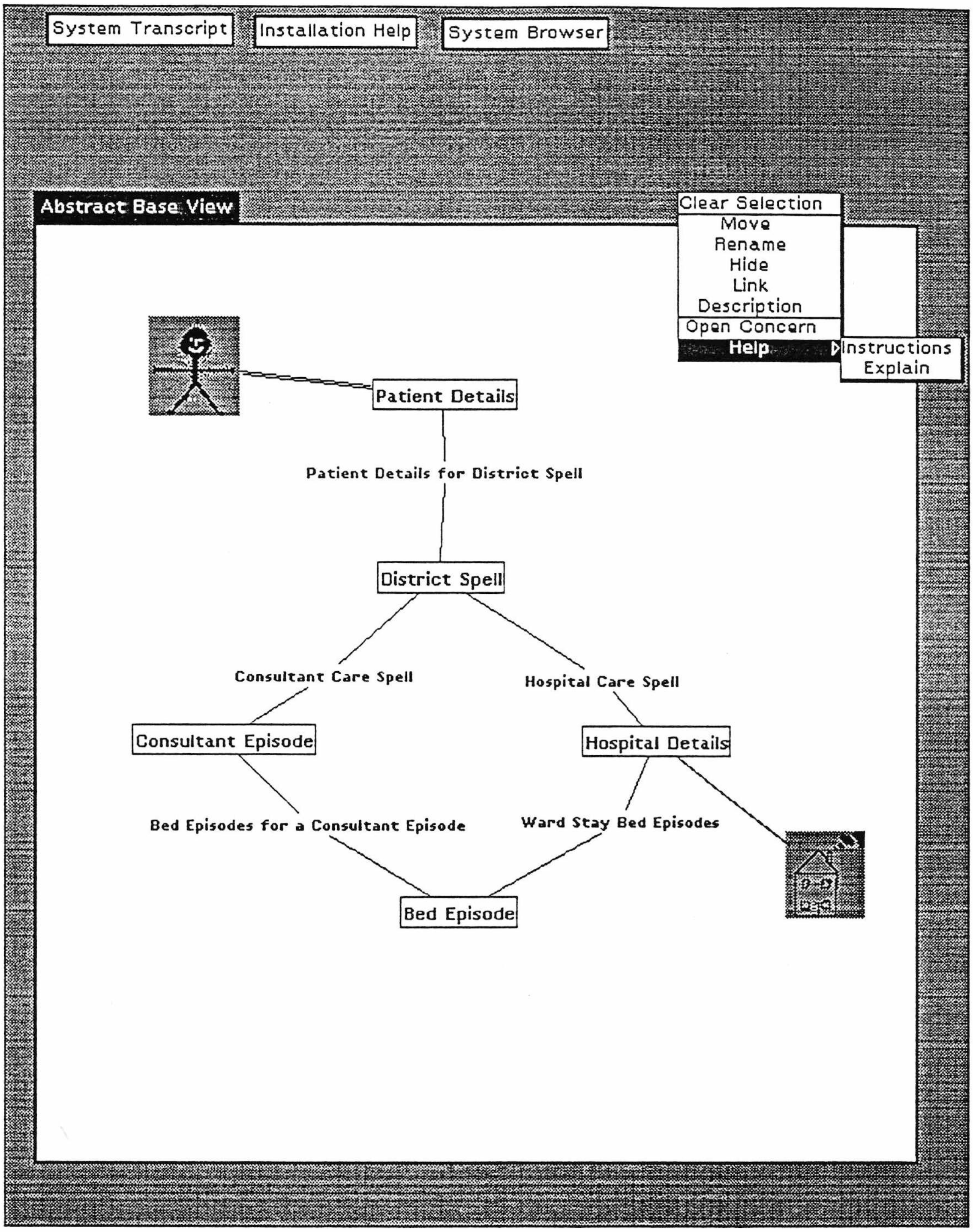


Figure 42 Initial QBC StartUp Presentation

convention is provided as a point and select device. In this view any number of objects can be selected at a time, being held in an *orderedCollection* instance variable in the view. The controller asks the view for the complete collection, processing them sequentially with the menu option selected.

Yellow Button (Pluggable - subclassed locally)

Clear Selection	Clear the selection made.
Move	Move the selected objects together at the cursor dragging them until any mouse button is clicked.
Rename	Give a new name to the selected screen objects.
Hide	Remove the selected objects from the *displayList* to a buffer for possible recovery.
Link	Create a link starting at the last selected object finishing on object with next mouse click
Description	Build a description object in area defined by user.
Open Concern	Generate a concern view from the areas of interest selected.
Help -> Instructions	Provide a list of directly available instructions applicable from this point.
-> Explain	Provide an explain view for selected object, showing internal descriptions which can be modified directly by the user.

The blue button functionality is inherited from the *standardSystemController* providing overall window management and is not repeated here.

6.6.3 Concern View

6.6.3.1 Description and Aims

The purpose of the concern view is to aggregate together all the information a user has about a particular domain into a graphically defined problem domain. This is done by the user generating a concern from the base list, specifically identifying those areas of interest that the real life concern may involve. To aid this discussion of concern and query specification, an example query of ICL DIS introduced in Chapter Three will be developed. This is a real world request as suggested by the District Information Manager in District D requiring the date of birth of all patients who died of heart disease categorised by consultant. The first stage in such a request is for the user to identify the main areas of interest held in the database which could be used to fulfil the request. These areas are selected from the base list, giving:

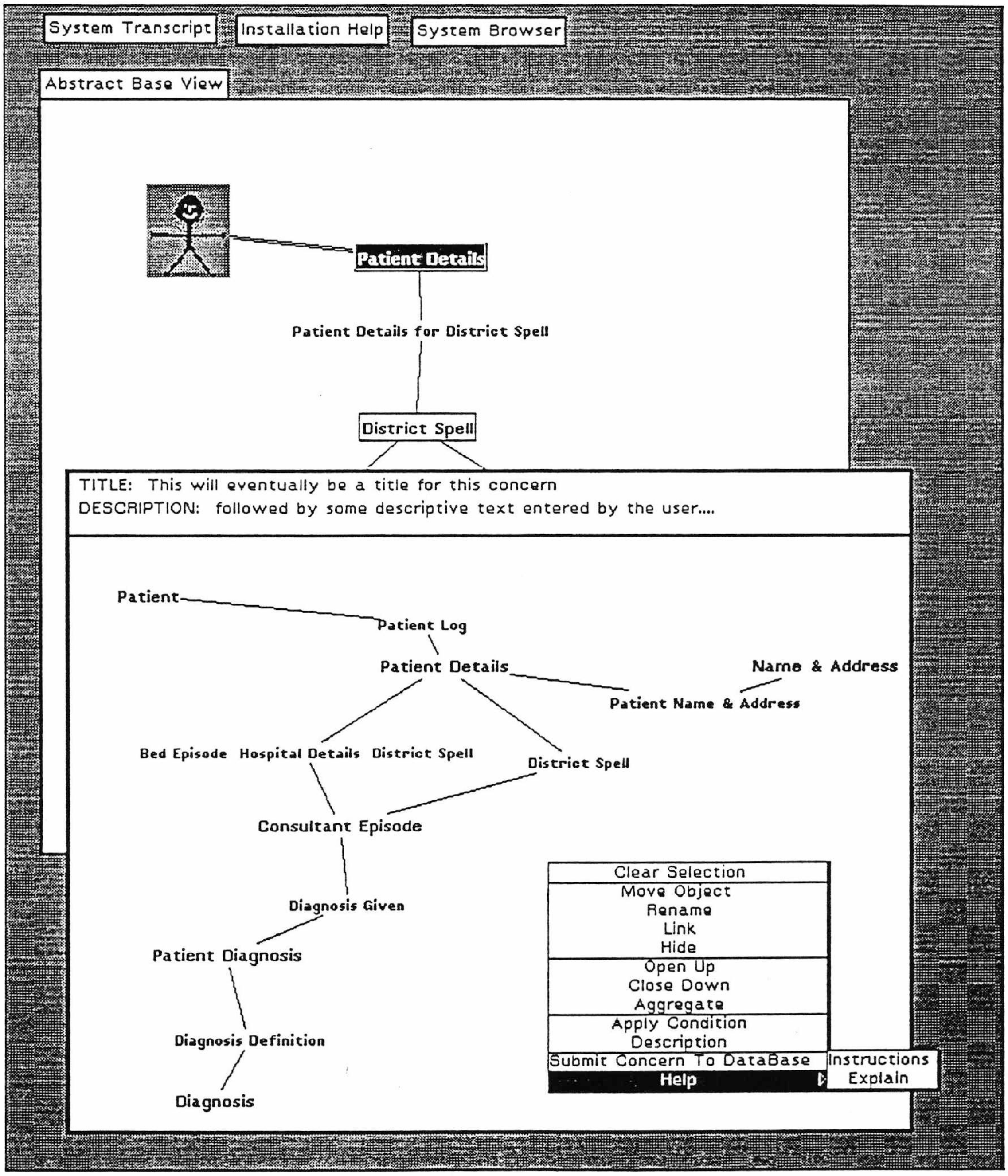
Patient-Details

Consultant-Episode

From this input, the application derives a concern view that is located on the display by the user. Within this newly opened view each area of interest is redisplayed in terms of the specific conceptual view represented, revised for screen clashes and relocated towards the view origin (top left corner). To provide a complete semantic map, the selected areas of interest are linked in terms of the available paths in the specific conceptual view based on the border nodes of each area of interest[b]. Each border is asked for its trunk pathings to every other border in the concern. For each unique path a macro linkage in installed into the concern. These are labelled for the user in abstract terms, that is: regarding the areas of interest they pass through. In addition multiple links which may have different paths but pass through the same

[b] Further detail of the process linking together areas of interest in a concern was discussed in detail in Chapter Five.

areas of interest are grouped together into the same link object. This reduces the number of links to be displayed, as well as providing a representation in terms the user is more likely to be familiar with. The initial concern view is presented in Figure Forty Three below.



TITLE: This will eventually be a title for this concern
DESCRIPTION: followed by some descriptive text entered by the user....

Figure 43 Concern for Example Query

Screen presentation is vital to user satisfaction and application utilisation. This area requires still further work to ensure that an acceptable screen presentation algorithm is built, for example to minimise line and screen object clashing.

A suggested approach would be to provide the screen structure in an initial oval shape from which the user must structure a coherent picture[238]. However such an approach requires user interaction and thought processes for each concern opened up, a possible disincentive to use. The implementation here has been biased towards the systems designers graphical layout in the base view, with each abstract entity capturing the spatial layout of its associated base objects.

With the concern open, the main task for the user is to tidy up this view, building a specific task based interpretation of the request. Objects can be added into the structure by the user to aid interpretation, for example description objects may be added providing text linked by pointer to any other screen object.

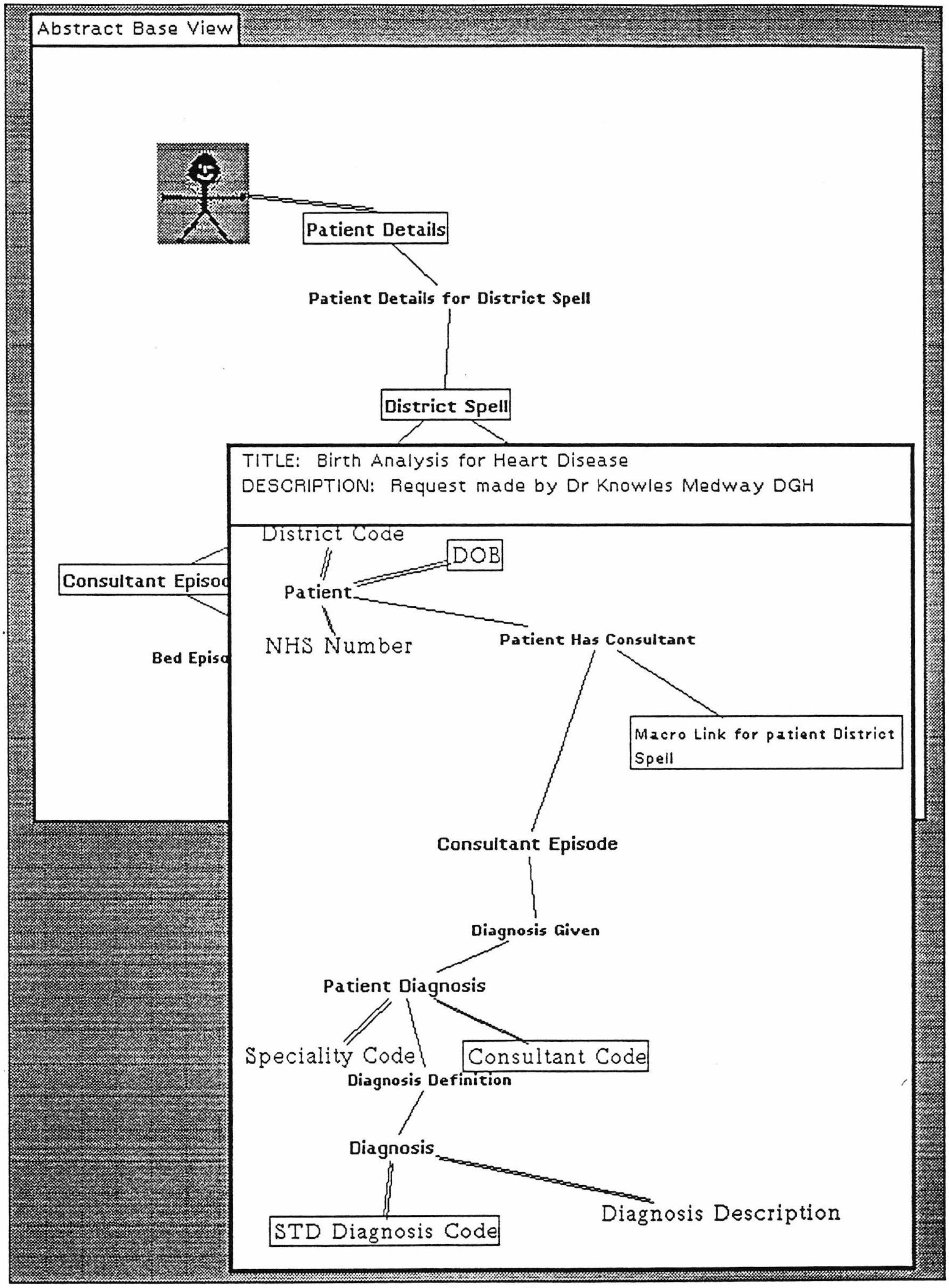
A second kind of screen object that may be added by the user, building a more complete interpretation is a macrolink, allowing the user to generate relationships between any two screen objects. In specifying the macrolink, the application captures a mapping of the possible paths through the generalised model but is constrained to only those areas of interest selected for the concern. Here it is assumed that links and possible paths outside the selected areas of interest are outside the scope of the problem domain and are therefore ignored. Such a situation is provided in Figure Forty Five. Here a user defined link is being used to simplify a concern structure removing unnecessary detail(left), bringing the presentation closer to a personal interpretation(right).

A development of this approach would have been to provide a further class of screen object by which the user could now start to develop personalised data set objects, ultimately building a localised database into the interface environment directly[239].

The approach taken here has been to give the user tools to manipulate the local problem domain, with the aim of providing a user visual interpretation of the information structure within the application for the particular task at hand. For example a refined version of the concern for the patient throughput analysis may be as seen in Figure Forty Six. From this stage the user may proceed to specify a query as discussed in Section 6.6.4 or collapse the concern, integrating this into the base view.

6.6.3.2 Internal Structure

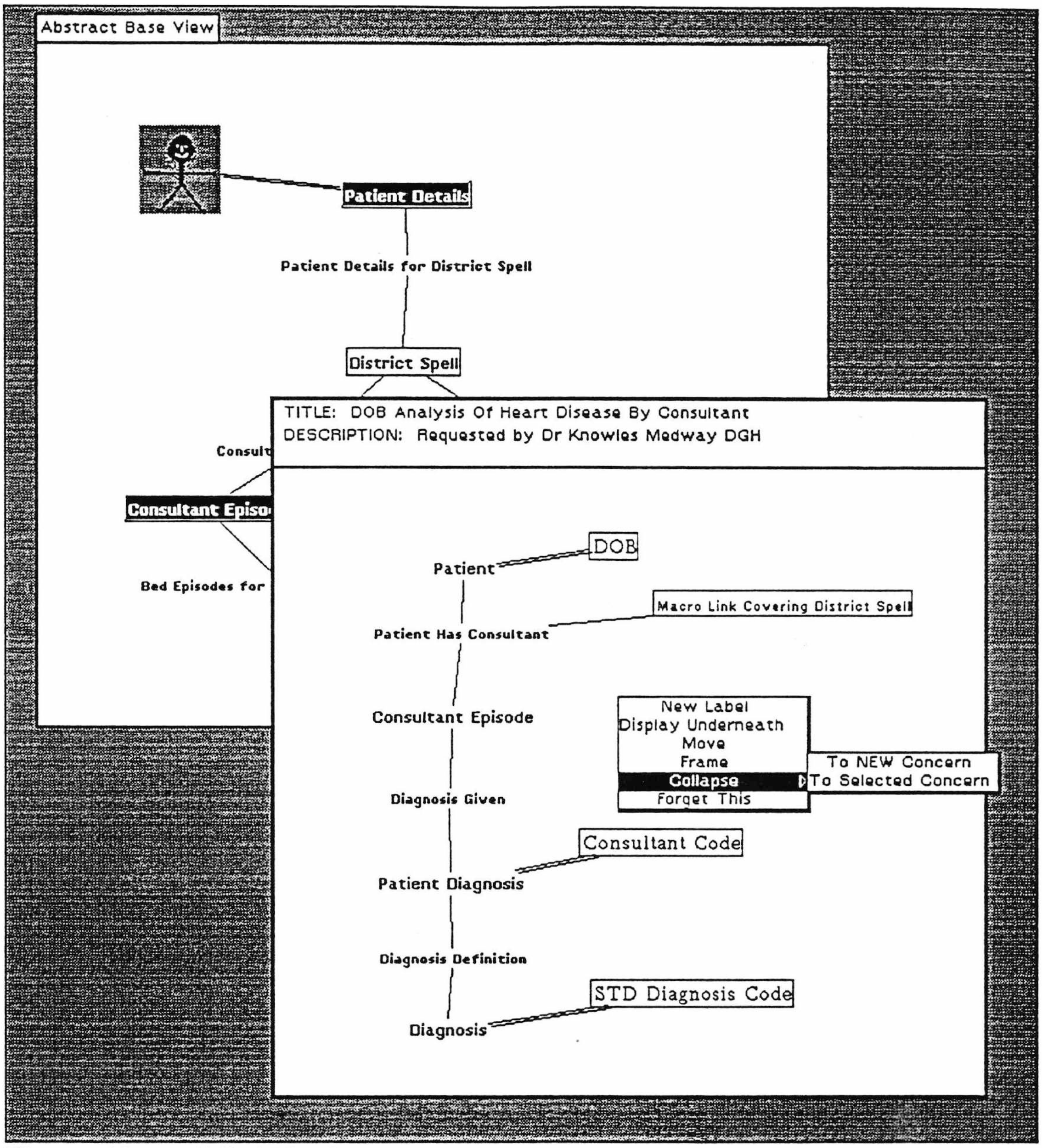
An opened concern is built from three views: a shell view, a text view and the *displayListView*, each being a MVC triad. The shell view is subclassed from *standardSystemController* providing a localised blue button menu, the *displayListView* is subclassed from *selectListController* providing localised yellow



TITLE: Birth Analysis for Heart Disease
DESCRIPTION: Request made by Dr Knowles Medway DGH

Figure 44 MacroLinkage and Description Boxes

menu functionality and the *textView* inherits from *stringController* providing a textual model based input with defined menu functionality. The model for the *displayListView* is an aggregation of displayLists from each area of interest.



TITLE: DOB Analysis Of Heart Disease By Consultant
DESCRIPTION: Requested by Dr Knowles Medway DGH

Figure 45 Tidied Up Concern

6.6.3.3 Menu Functionality

Three mouse buttons are provided, the protocol for each being inherited from the *mouseMenuController* structure. The red(left) button following Smalltalk convention is provided as a point and select device depending on the view, for example within the text view as a pointing device locating the text cursor, whereas in the main view acting as a selection device. The blue and yellow button operations are 'plugged' with local definition and are described here.

Yellow Button (Plugged with local definition)

Clear Selection	Clear the current selection.
Move	Move the selected objects together at the cursor dragging them until any mouse button is clicked. Move
Rename	Give a new name to the selected screen objects.
Link	Create a link starting at the last selected object finishing on object with next mouse click.
Hide	Remove the selected objects from the *displayList* to a buffer for possible recovery.
Open Up	Display the attributes relating to an entity.
Close Down	Remove from the display the attributes associated with the selected entity.
Aggregate	Aggregate together the selected entities into a single screen representation.
Apply Condition	Apply a condition/instruction to an entity or attribute
Description	Build a description object in area defined by user.
Submit Concern To Dbase	Process a concern as a database query.
Help -> Instructions -> Explain	Provide a list of direct available instructions. Provide an explain view for selected object, showing internal descriptions which can be modified directly by the user.

Blue button functionality is concerned with manipulation of the view as a whole. This has been subclassed to allow the view to be stored into a button object within the base list for further use. The functionality provided is as follows:

Move	Drag the view to another location on the screen maintaining proportions.
Frame	Alter the size and shape of the view.
Under	Push this view to the bottom of the stack of views on the display.
Collapse to Selected Concern	Collapse this concern leaving the same representation in the base view.
Collapse to New Concern	Collapse this concern to a new representation in the base view.

6.6.4 Query Processing

6.6.4.1 Description and Aims

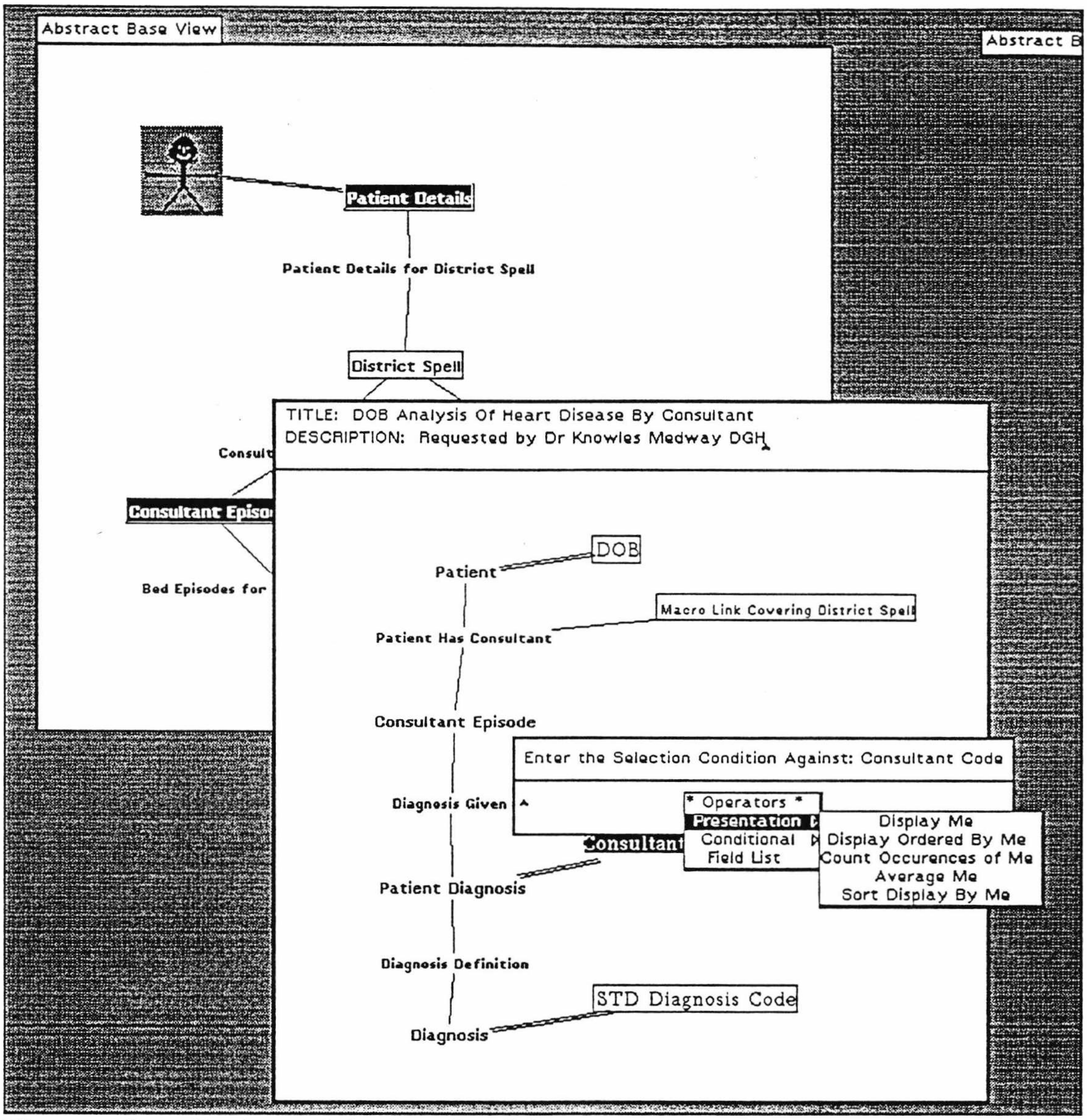
The aim of the application has been to enable the user to specify a request for information within an environment tailored, not only to an overall personal interpretation of the information structure, but also specifically related to the task at hand. The concern provides this environment. In this section we describe how a query is specified within a concern and eventually submitted to the database.

To represent a query, the aim is for the user to express requests onto individual entities or attributes that are subsequently processed once the user is satisfied with the built representation. The output from this process will be a query ready for processing into a final syntax ready for database submission.

To express a request onto a given attribute the menu operation 'Apply Condition' provides access to an instance variable in the object against which the user may enter selection text, either directly for example if an experienced user, or by following the menu operations available[240]. For the query patient analysis query specified in 6.6.3, the following conditions are to be applied:

Entity	**Attribute**	**Setting**
K-Diagnosis	Diagnosis	= 'Heart Disease'
K-Patient-Diagnosis	Consultant-Code	SORT-BY
K-Patient	Date-Of-Birth	PRINT

For example setting the condition that the query output be sorted and ordered by the Consultant-Code would be achieved as follows: (using the menu structure, alternately it is possible to type directly the selection condition) Submitting the query to the database option from the yellow button, processes the *displayList* in the current concern, forming a path between those objects with selection conditions applied. In doing so, duplicate paths may arise for two reasons. Firstly there may be duplicate paths presented on the screen through the developed concern. If this is the case, then a graphical choice has to be made by the user indicating the preferred path. For example, Figure Forty Seven shows there are two paths available connecting those objects with selection conditions applied (boxed presentation). A graphical choice is required to ensure the correct navigation is performed through the data structure. The possible paths are presented within the concern, preserving context rather than extracted out into a separate and therefore isolated view.

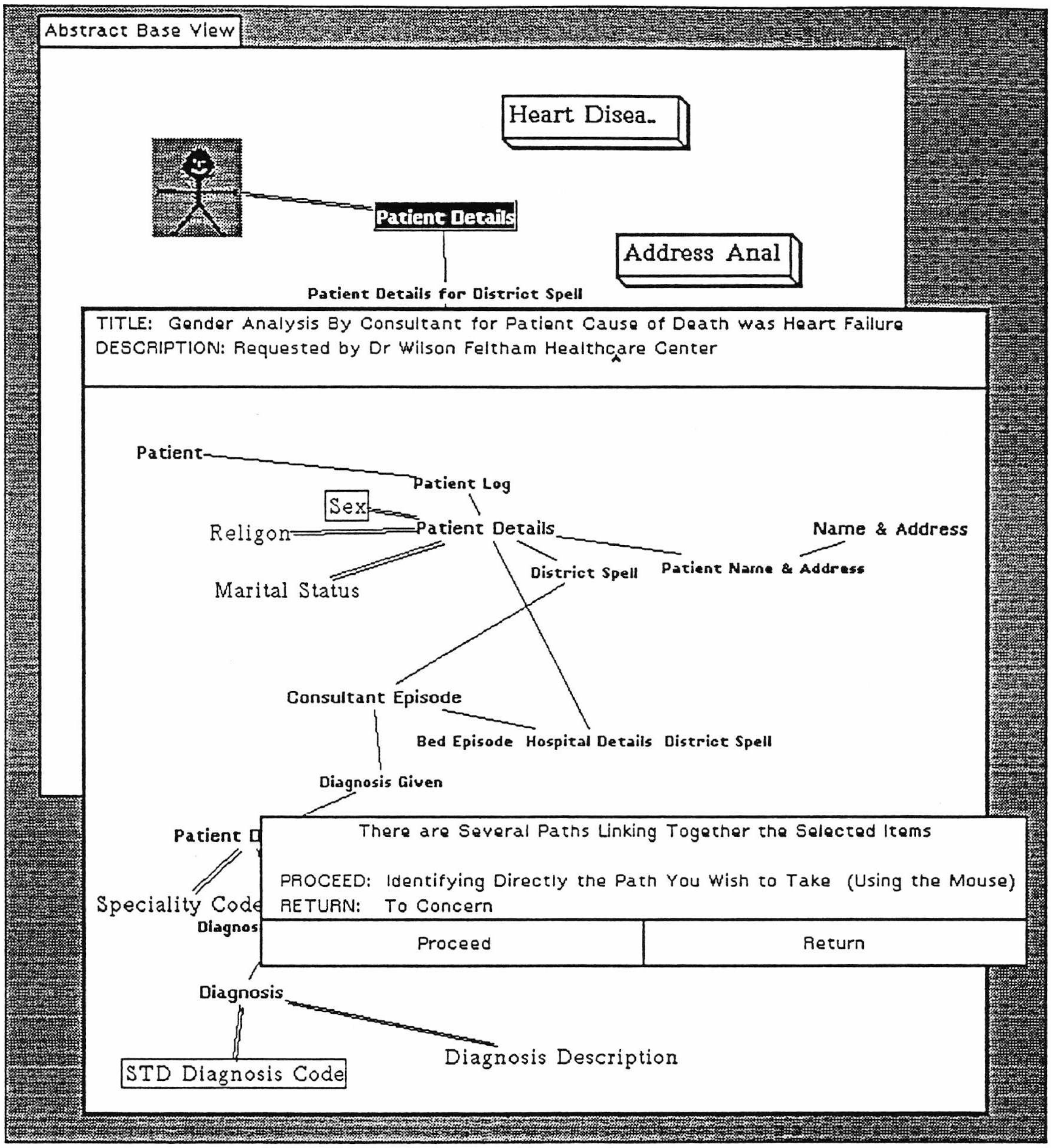


TITLE: DOB Analysis Of Heart Disease By Consultant
DESCRIPTION: Requested by Dr Knowles Medway DGH

Enter the Selection Condition Against: Consultant Code

Figure 46 Setting a Selection Condition

Each additional link applied connecting areas of interest or the additional user defined macroLinks hold a macro path in terms of the abstract model. This path is therefore expressed regarding areas of interest in which there may be multiple paths when mapped to the base model. If a required path includes such a subpath then the link has to be opened up and a required path stipulated. A user choice is required from the available possibilities. These are redrawn as shown in Figure Forty Six from which the user identifies key entities that the path must include.



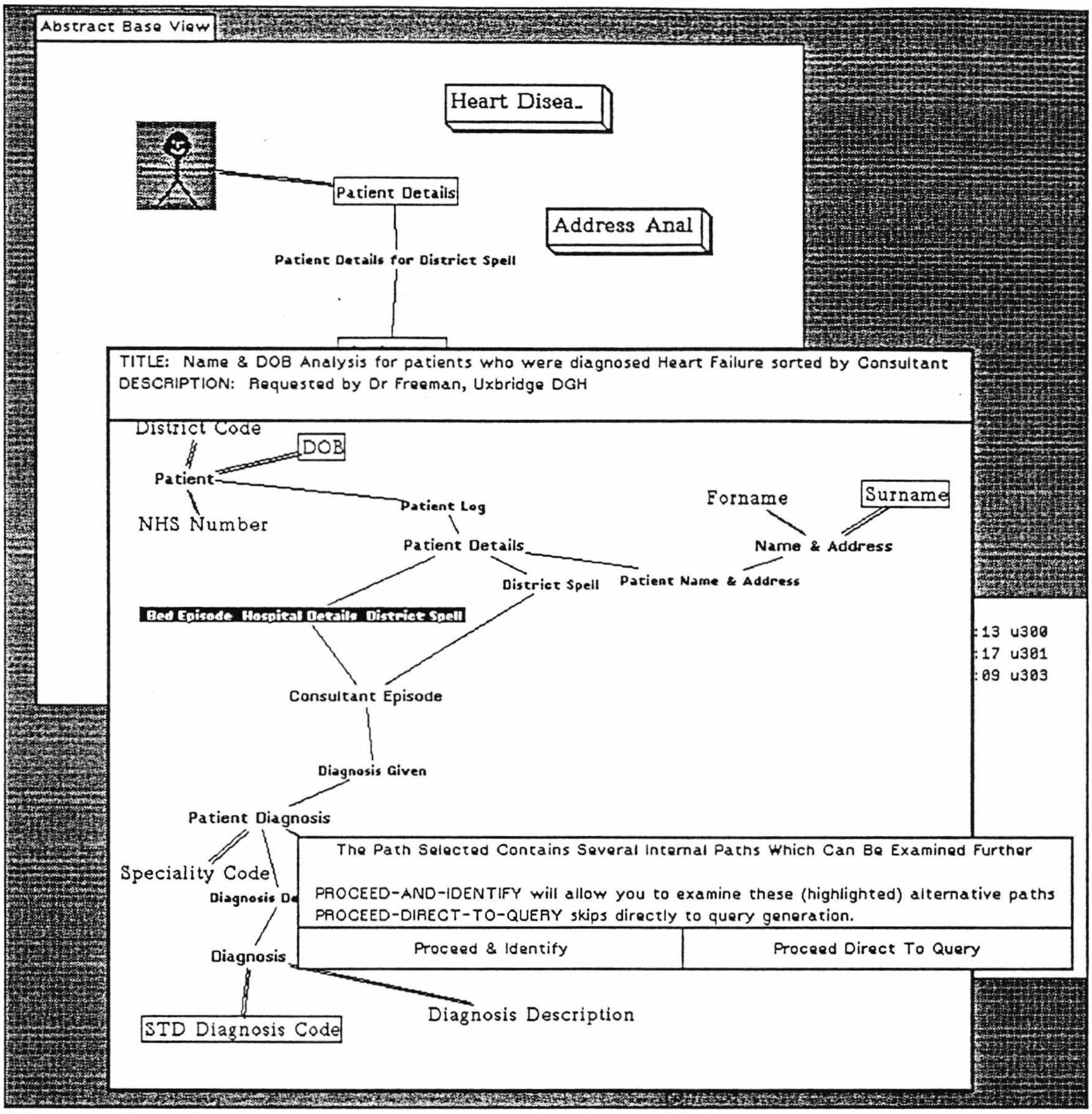
TITLE: Gender Analysis By Consultant for Patient Cause of Death was Heart Failure
DESCRIPTION: Requested by Dr Wilson Feltham Healthcare Center

There are Several Paths Linking Together the Selected Items

PROCEED: Identifying Directly the Path You Wish to Take (Using the Mouse)
RETURN: To Concern

Figure 47 User Participation is Required When Have Multiple Paths

Once the pathing has been defined (if necessary) each object with a condition is processed and the query produce. From this point the output is ready for processing into any required query language syntax.



TITLE: Name & DOB Analysis for patients who were diagnosed Heart Failure sorted by Consultant
DESCRIPTION: Requested by Dr Freeman, Uxbridge DGH

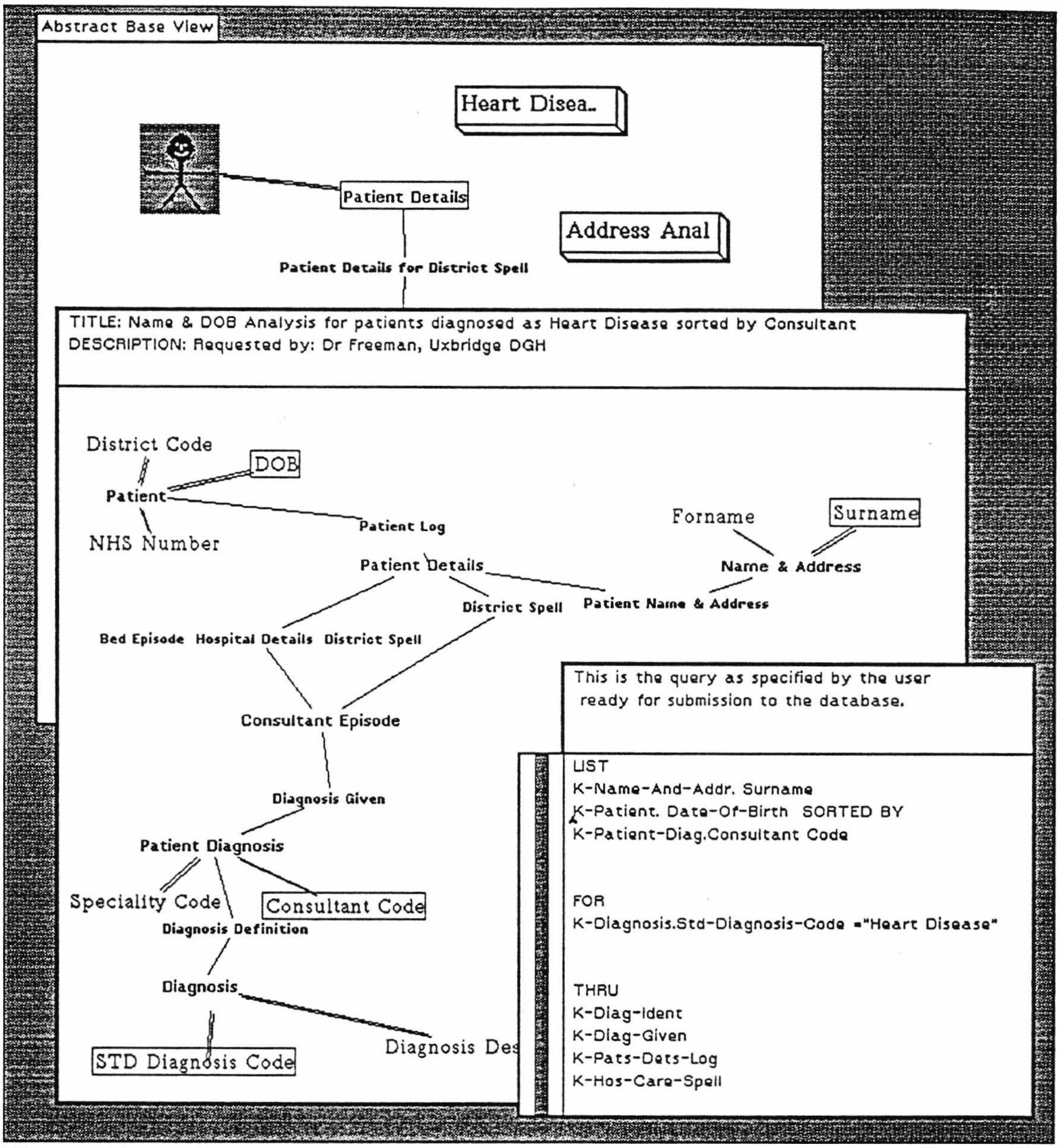
The Path Selected Contains Several Internal Paths Which Can Be Examined Further

PROCEED-AND-IDENTIFY will allow you to examine these (highlighted) alternative paths
PROCEED-DIRECT-TO-QUERY skips directly to query generation.

Figure 48 User Interaction Required for MacroLinkages

6.6.5 Summary

This side of the application has aimed to provide an environment in which the user can manipulate the presented semantic structure to suit the final end user's own interpretation of the information held within the application. This environment has two key components: the base and concern views. The base view allows overall graphical interpretation of the information within the structure. This is primarily at an abstract level consisting of areas of interest,



TITLE: Name & DOB Analysis for patients diagnosed as Heart Disease sorted by Consultant
DESCRIPTION: Requested by: Dr Freeman, Uxbridge DGH

This is the query as specified by the user ready for submission to the database.

```
LIST
K-Name-And-Addr. Surname
K-Patient. Date-Of-Birth  SORTED BY
K-Patient-Diag.Consultant Code


FOR
K-Diagnosis.Std-Diagnosis-Code ="Heart Disease"


THRU
K-Diag-Ident
K-Diag-Given
K-Pats-Dets-Log
K-Hos-Care-Spell
```

Figure 49 Query Output Complete

relationships between these areas, past concerns and user defined areas of interest.

The concern component provides a mechanism by which the user can generate personal areas of interest, as well as generate queries of the data base itself. Personal areas of interest are provided by manipulating a selection of areas of interest within a single concern and storing as a surrogate area of interest.

Alternatively the concern is used as a mechanism to aggregate together all the information a user has about a particular problem domain into an enclosed graphically defined domain. Doing so builds up a personalised but also task oriented environment in which the user may express a query. Such a concern can then be stored back onto the base list, capitalising knowledge within that particular area of the information structure for further interpretation.

6.7 Overall Summary

The aims of the prototype set in Chapter Five relate to the general provision of an interpretative environment in which the individual user may incorporate both personal, organisational and task based information into the viewing and interrogation mechanisms of the information resource. This chapter has provided a detailed description of the implementation of the prototype and principles in an attempt to meet these goals.

Several other systems have attempted to provide a user interface built upon some form of semantic model, usually stipulated by the user at some point[197,241,199]. This approach has linked the query specification and interpretation of information within the information structure back to the specification of that structure, providing a complete approach to database implementation where the user is not required to explicitly specify the semantic model, only the incremental changes. The key backbone of the application design has been implemented. All the menu functionality is provided but there remains areas to be refined and enhanced further, for example little error checking is provided.

What has been achieved is a prototype linking both the systems designer and the end user. The systems designer is provided with a graphically driven environment from which the mapping between the generalised and the specific is captured and used as a basis for end user interaction. DDL code is generated as a byproduct. The abstract conceptual view forms the basis for the end user environment acting as an initial starting point for interpretation of the information held within the application structure. Upon this is built a personal interpretation of specific areas of information or tasks which are then capitalised back into the system to aid future interpretation.

CHAPTER SEVEN

Conclusions and Further Work

Contents

7.0 Conclusion

The role of computing technology within the organisation has filtered through the shop floor into routine office functions, and now most recently into the core of the organisation: central management decision support. Through this development, bureaucratic reporting processes have been reassessed and transformed into sources of organisational advantage. The organisation has invested considerable resources into this process, pursuing perhaps an unquantifiable benefit: improved decision making.

Like any other asset in the organisation, the maintenance and provision (often termed Information Resource Management) of this asset has to be justified in terms of its effectiveness and efficiency. Using these criteria as a means of judging the success of the information resource however remains problematic, especially the determination of value based output, for example improved decision making. Several authors have suggested the surrogate measure of utilisation, but this remains a complex variable to evaluate. This work has concentrated on one aspect of utilisation that is increasing in importance: user accessibility.

The convergence of increased processing power, mass storage technology and physical accessibility to diversified sources of information, has facilitated generation of increasingly large and detailed organisationwide applications. A concurrent effect as the price of this technology has fallen into individual line management budgeting, has been the type of staff that are coming into contact with these computer based applications has diversified, rapidly encompassing non computer literate personnel. These opposing components of utilisation can only intensify as the technology progresses, meeting through the process of user accessibility.

To investigate this issue, a field study was carried out within one Region of the National Health Service at District Health Authority level. The historical role of information provision within the Service has been analyised to the present, moving from a centralised, bureaucratic overhead to a localised source of management information. Under increasing cost pressure, the utilisation of these information systems is coming under increasing scrutiny, not least because they can often be perceived as diverting resources away from patient care. The study was carried out in the Summer of 1989, investigating utilisation, but particularly the role of accessibility to this resource.

Each District has developed an intermediary role for information provision, cushioning the final user from the information resource by an extra stage of interaction. Two stages of accessibility were therefore identified from the study. The fist stage identified was between the user, for example clinician or UGM and the intermediary function. The key variable of user awareness was identified, especially when comparison was drawn with the reference library model where the user had no open shelf browsing facility. The mechanisms in use to promote user awareness varied considerably, a strong correlation was identified between those Districts who actively promoted the information function, that is either through wide provision of training in information availability and use or in making information directly available, for example through information shops, browsers etc and overall utilisation of the information resource.

The variable of user awareness plays an important part from the Information Manager's viewpoint also. Little or no contextual training has been provided to Information Management (the intermediary), the training provided concentrated exclusively on the procedural elements in query generation, rather than query generation and application within the organisation. In practical day to day terms, information management has been forced to rely heavily on background knowledge of the Health Services. Those managers who have not come from a HS background have encountered significant problems of information interpretation, the identification of information relevance and information validity checking.

For the Information Manager, utilisation has been identified as contingent upon accessibility, that is an aggregation of interface and data model complexity. A single rigid interface (regardless of user background, demands of the system or individual knowledge), was provided for all users. In addition the data model provided was a past, fixed interpretation by a systems analyst of the information resource provided. The size, structure and detail of this datamodel provided a significant level of complexity, with the user having to be aware of the entire data structure, the current location and explicit routes through the structure to build an effective query. Those managers who have come from a computer science background were better equipped to handle these problems. Managers from a Health Service background reported significant difficulty, the combination of a complex data model and poor interaction tools provided a significant disincentive to use the information resource. In practical terms, the Information Manager was faced with a complex data model and complex access tools with which to provide the bridge to the user community.

The interface environment of an application *is* the application from the user's perspective. The interface provides all of the application that the user actually sees and is aware of. However provision of this user interface has often been regarded as an afterthought from the computer science engineering discipline. At its best, the user was assumed to be a quantifiable component like any other module within the application. Through the 1980's the multidisciplinary field of Human Factors has developed, promoting user centred rather than system centred human computer interaction.

To address the two key issues raised from the field study, a Human Factors approach has been combined with components from the data modelling process, promoting interpretation and user control of the application. The basis of the prototype is that effectiveness of the user interface to a complex information resource can be improved through the capture and user provision of the generalised conceptual model as a basis for interpretation.

The aim here was to provide an interpretative environment in which the user can impose his/her own personal interpretation of the information content of the resource. This would be built up by combination of organisational knowledge, general information, task related information and past interactions within a completely graphical WIMPS 'WYSIWY can get hold of and fiddle with' environment.

To meet this challenge, a prototype called Query By Concern(QBC) was built. QBC provides a graphical interpretative environment, built upon the generalised conceptual model as captured in graphical format from the systems modelling stage in the information resource specification and design. QBC is composed of two parts: systems modeller and end user parts. For the systems modeller, the generalised and specific conceptual models are graphically developed with the logical data model generated in code form as a by product. For the second part, the end user builds up an interpretation of the information resource based on the generalised conceptual model, captured in a graphical form from the systems modeller.

The aim is for the user to fashion his/her own graphical interpretation, taking this only as the base start up interpretation. This is achieved principally through the development of 'concerns'. A concern provides a graphical problem domain in which the user can express a localised interpretation of a particular part (or of the whole) of the semantic structure, using graphical tools such as move, hide and rename. The user can then express a query within this personal and task tailored interpretation of the information resource. This concern may then be integrated into the overall graphical interpretation of the semantic content of the database.

In this manner an interpretative environment has been developed whereby each user can interpret the information resource both in terms of his/her overall perspective of the system at a macro level but also at a micro or individual concern level. Within this component, several approaches to programming methodology and style of interface environment have been considered. The underlying basis is for a strong object orientated environment closely linking the code, data structures and the user application design into a functioning prototype. From the user design perspective, several approaches to the user interaction style have been identified, with the direct manipulation in a graphical based environment approach adopted. The prototype, developed in Smalltalk-80 has combined this direct manipulation user environment style with the object orientated paradigm. The network paradigm of DIS has formed the example basis of both the prototype development and thesis presentation. Following the modularity available in object oriented software development, equal opportunity exists for the development of the relational and hierarchial data models within the prototype.

The work demonstrates how an interface designed and developed from the users perspective can go a long way in helping to overcome the complexity issue in accessing a complex information resource. In overall terms the contribution made by this work has been to tackle the user interface bridge in an applied manner. The developed prototype demonstrates how the generalised conceptual interpretation of the information structure, which is available as an unused aspect of the data modelling process, can be linked with the end user environment to provide a platform for successful user interpretation of the database

7.1 Enhancements to the Prototype

Clearly QBC is far from ideal. Many improvements could be made to both the structure of the prototype and the user interface environment. Smalltalk-80 provides a basic user interface toolkit which was used as a development platform rather than as an experimental basis developing new unique graphical structures. The graphical presentation of the interface therefore remains comparatively simple, as the work focused on the interaction style, rather than presentation outcome. As a result, much work remains in this area for QBC, for example when opening up a concern a more effective algorithm to minimise screen clashing of displayed objects is needed. Another example is the whole area of colour and its use to highlight importance, provide arrangement to a structure, and indicate status within the environment.

Within the prototype itself, further implementation of some remaining menu functionality is required:

- HELP menu option has not been implemented although each object carries instance variables for such information

- AGGREGATE menu option has been implemented but further development work remains

- UNDO option is provided but not implemented

- CUT/COPY/PASTE is a logical next step in implementation, providing for the cut/copy/pasting of graphical screen objects both within a concern and between concerns. However problems of model integrity remain to be addressed for this issue.

- ICONS are provided depicting core categories of people, places(buildings) and reports. An initial aim was to provide these as a further layer of abstraction for concern building. Further development is required to implement this requirement.

- scope remains to develop the data modelling tools available to the systems modeller, for example explicit identification of record primary keys in a graphical manner.

One of the key presentation issues left unaddressed by ISIS, SNAP and QBD, which is also left partly unaddressed by this work, is the problem of graphical presentation size and subsequent overspill onto additional screens. The best that is available is the use of slider bars, but which remain unsatisfactory, providing an unacceptable increase in mental load for the user as the structure disappears off the working screen. This issue is addressed only indirectly through the development by the user of successive graphical aggregations, which however require increasing user specification and experience to achieve. This important issue therefore remains unaddressed.

Through the thesis, the prototype has been presented using the DIS networked data model. The prototype has been developed in a highly modular fashion, requiring only module substitution to provide either the hierarchial or relational modelling paradigm. With sufficient time, these modules would have been developed and presented.

7.2 Future Directions

It is only in recent years that the important role of the user interface and provision of HCI have been recognised in successful application development. This work has aimed to further this process in a field which is expanding rapidly. Swift advances have been made in the development of hypermedia and multimedia systems, building progressively larger, complete sources of information which include high resolution pictures, graphics and sound. Ultimately these approaches will encounter similar problems of structural complexity discussed in this work. For example, the already well established 'getting lost' problem is just the beginning, as new forms of information and media are integrated into increasingly complex information bases.

With any form of HCI, at some point the user is still required to labour at the interpretation, that is the gulf of evaluation, for example building or maintaining the interpretative environment. Human Factors on its own is therefore not the complete answer to the question of accessibility and utilisation. Another area within computer science itself, may hold the key: expert systems.

Expert systems can be combined with human factors in two ways. The first application of expert systems would be as part of the user interface environment, assisting the user to express a query, for example combined with natural language interface design. Secondly, expert systems may be provided as a tool to assist evaluation of the data output from a database query, for example in terms of maintaining the context of the data retrieved, and its conversion to information.

Using these tools, regardless of how user friendly and 'intelligent' an interface environment to an application may become, computer science has always assumed that such systems are effectively installed in a social vacuum. This is not the case, with any system installation provoking organisation, political and social issues. The development of Human factors is an encouraging sign towards an applied perspective of application design, drawing the integration of multiple disciplines together towards the aim of a 'successful' organisation resource. For the future therefore, encouraging computer science away from the Model T premise of application production is vital, especially as new demands and challenges are made of the information resource, for example the development of group decision support systems[242].

Bibliography

1. Information Technology: A Bibliography
   Department of Industry
   London 1982

2. Mackenzie D
   Why The Social Aspect of Science & Technology Is Not Just An Optional Extra
   ACM SIGCAS Computing Society Winter 1986 Vol 15 No 4 pp2-N6

3. Kraut R, Dumais S, Koch S
   Computerisation, Productivity & Quality of Work Life
   Communications of the ACM Feb 1989 Vol 32 No 2 pp220-238

4. Cooley MJE
   Architect or Bee? The Human Price of Technology
   Hogarth Press, London 1987

5. Bucannan DA, Boddy D
   Advanced Technology and the Quality of Working Life: The Effects of Word Processing on Video Typists
   Journal Of Occupational Psychology 1982 No 55 pp1-11

6. Strong G
   The Database Tool and the Information Assembly Line
   Computers & Society Spring 1986 Vol 16, No 1 pp14-16

7. Vlist R
   Information Processing Activities & Operational Decision Making: A Case Study of Consequence
   Information & Management 1989 Vol 16 pp219-225

8. Noble D
   Social Choice In Machine Design: The Case for Automatic Controlled Machine Tools In: Timbalist T (Ed)
   Case Studies In the Labour Process
   Monthly Review Press New York 1979

9. Wiliamson B
   Shop Floor Politics
   Gower Publishing 1986

10. Eason KD
    Managing Technological Change
    In: Paton R (ed)
    Organisations: Cases, Issues, Concepts
    Harper & Row Publishing 1984

11. Sumner M, Klepper R
Information Systems Strategy & End User Application Development
DataBase Summer 1987 Vol 18 No 4 pp19-30

12. Carlson C
Information Management Approach & Support To Decision Making
Information & Management 1988 Vol 15 pp135-149

13. Necco C, Gordon C, Tsci N
The Information Center Approach For Developing Computer Based Information Systems
Information & Management 1987 Vol 13 pp95-101

14. Sipior J, Sanders GL
Definitional Distinctions & Implications For Managing EUC
Information & Management 1989 Vol 16 pp115-123

15. Rockart J, Flannery S
The Management of EUC
Communications Of The ACM Oct 1983 Vol 26 No 10 pp776-784

16. Long RJ
The Application Of MicroElectronics to the Office: Organisational and Human Implications
In: Piercy N (ed)
The Management Implications of New Technology
Ch6 pp95-121 Croom Helm Nichols Publishing Co 1984

17. Burd S, Kassicieh S
Logic Based Decision Support For Computer Capacity Planning
Information & Management October 1987 Vol 13 No 3 pp125-133

18. Preedy D
The Theory & Practical Use of Executive Information Systems
International Journal Of Information Management
1990 Vol 10 pp96-104

19. Beheshtian M
Strategies For Managing User Developed Systems
Information & Management 1987 Vol 12 pp1-7

20. O'Donnell D, March S
End User Computing Environments - Finding A Balance Between Productivity & Control
Information & Management Sept 1987 Vol 13 No 2 pp77-84

21. Zijker A
Setting The Scene for Information Resource Management
Information & Management 1988 Vol 15 pp79-84

22. Zmud RW, Boynton AC, Jacobs GC
The New Information Economy: A New Perspective for Effective IS Management
DataBase Fall 1986 Vol 18 No 1 p17-25

23. Molholt P, Hohenberg PM
What Price the information Age? A Model For Fair Payment For the Use Of Information
In: Zunde P, Agrawal J (Eds)
Empiciral Foundations of Information & Software Science IV: Empirical Methods of Man Machine Interfaces
1988 North Holland pp479-486

24. Taylor RS
Value Added Processes In Information Systems
Information Services & Use June 1984 Vol 4 No 3 pp127-146

25. Doll W
Information Technology Strategic Impact On The American Travel Service Industry
Information & Management 1989 Vol 16 pp269-275

26. Best DP
The Future of Information Management
International Journal Of Information Management Vol 8 No 1 pp13-19

27. Trauth E
The Evolution of Information Resource Management
Information & Management 1989 Vol 16 pp257-268

28. Nolan R
Managing The Computer Resource: A Staged Hypothesis
Communications Of The ACM 1973 Vol 16 pp399-405

29. Brussaard BK
IRM In The Public Sector
Information and Management 1988 Vol 15 pp85-92

30. Lewis D
Information Management: The Way Ahead
International Journal Of Information Management Vol 8 No 1 pp61-68

31. King JL, Kraemer KL
IRM: Is It Sensible, Can It Work?
Information & Management 1988 Vol 15 pp7-14

32. Grover V, Leener A, Sabhenual R
Recognising the Politics of MIS
Information & Management 1988 Vol 14 pp145-154

33. Carrilio T, Karner J, Moretto A
MIS: Who Is In Charge?
Journal Of Social Work Sept 1985 Vol 66 pp417-423

34. Evans G, Riha J
Assessing DSS Effectiveness Using Evaluation Research Methods
Information & Management 1989 Vol 16 pp197-206

35. Swanson EB
Measuring User Attitudes In MIS Research: A Review
Omega: The International Journal of Management Science
1982 Vol 10 No 2 pp157-165

36. Raymond L
Validating & Applying User Satisfaction As A Measure of Success in Small Business Organisations
Information & Management 1987 Vol 12 pp173-179

37. Ives B, Olson M, Baroudi J
The Measurement of User Information Satisfaction
Communications of the ACM Vol 26 No 10 pp785-793

38. Igbaria M, Nachman S
Correlates of User Satisfaction With End User Computing
Information & Management 1990 Vol 19 pp73-82

39. Rivard S, Huff SL
Factors of Success For End User Computing
Communications of the ACM 1988 Vol 31 No 5 pp552-561

40. Igbara M, Pavri F, Huff S
Microchip Applications: An Empirical Look At Usage
Information & Management 1989 Vol 16 pp187-196

41. Rushinek A, Rushinek S
What Makes Users Happy?
Communications of the ACM July 1986 Vol 29 No 7 pp594-603

42. Zmud R
Individual Differences In MIS Success: A Review of the Empirical Literature
Management Science Oct 1979 Vol 25 pp966-979

43. Mykytyn P
An Empirical Investigation of DSS Usage & Users Perception of DSS Training
Information & Management 1988 Vol 14 pp9-17

44. Bailey J, Pearson S
A Tool For Measuring & Analising Computer User Satisfaction
Management Science 1983 Vol 29 No 5 pp530-539

45. Trice A, Treacy M
Utilisation As A Dependent Variable In MIS Research
DataBase Winter 1988 Vol 19 No 3. pp33-42

46. Fuerst W, Cheney P
Factors Affecting the Perceived Utilisation of Computer Based DSS in the Oil Industry
Decision Sciences 1982 Vol 13 No 4 pp554-569

47. Jagodzinski P, Clarke D
Criteria and Techniques for the Objective Measurement of Human Computer Performance
In: Zunde P, Agrawal J (Eds)
Conference Proceedings of Empirical Foundations of Information & Software Science IV: Empirical Methods of Evaluation of Man Machine Interfaces 1988 North Holland pp103-122

48. Nielsen J, Molich R
Heuristic Evaluation of User Interfaces
CHI'90 Proceedings April 1990 ACM pp79-84

49. Alavi M Henderson J
An Evolutionary Strategy for Implementing A Decision Support System
Management Science November 1981 Vol 27 pp1309-1323

50. Rice R, Shook D
End User Computing: Access, Usage and Benefits
Proceedings of the American Society For Information Science
1986 Vol 23 pp265-270

51. Waguespack LJ
Software Complexity Assessment & Human Machine Evaluation
In: Zunde P, Agrawal J (Eds)
Empirical Foundations of Information & Software Science IV in: Empirical Methods of Evaluation of Man Machine Interfaces
1988 North Holland pp187-204

52. O'Reilly
Variations In Decision Makers Use Of Information Sources: The Impact of Quality & Accessibility Of Information
Academy Of Management Journal 1982 Vol 25 No 4 pp756-771

53. Bozeman B, Shangraw R
Computers as a Public Management Decision Tool
Knowledge: Creation, Diffusion, Utilisation Dec 1988 Vol 10 No 2 pp111-139

54. Department of Health & Social Security
Table 7.33 Family Practioner & Dental Services (UK)
Health & Personal Social Services Social Trends 21 1991

55. Department of Health & Social Security
Table 7.39 Manpower In The Health & Personal Social Services (UK)
Health & Personal Social Services Social Trends 21 1991

56. Consultative Council on Medical and Allied Services
Interim Report on the Future Provision of Medical and Allied Services (Chair: Lord Dawson)
HMSO London 1920

57. Ministry of Health
Hospital Survey
HMSO London 1945

58. Ministry Of Health
A National Health Service
Cmnd 6502 HMSO London 1944

59. Ministry of Health
National Health Service Bill
Cmnd 6761 HMSO London 1946

60. Willkin K
HC Debates vol 438 Col 428 16th Mar 1944
Introducing the debate on the 1944 NHS White Paper

61. Local Government (Emergency Provisions) Act 1923
Ch35 Public General Act 15/6 Geo V 1924-5 Vol 2

62. Poor Law Act (Amendment) 1934
Public General Acts Ch 59 Geo V

63. Lunacy Act 1890
Public General Acts Ch5 53/54 Vict

64. NHS Amendment (Amendment) Act 1951
Ch 93 12,13 & 14 Geo pp1911-1943

65. Ministry of Health
Report of the Committee of Enquiry into the Cost of the National Health Service
HMSO Cmnd 9663 January 1956

66. Ministry of Health
First Report of the Joint Working Party on the Organisation of Medical Work in Hospitals Chair: Sir G Godber
HMSO 1967

67. Medical Services Review Committee
A Review of Medical Services in Great Britain
Chair: Sir A Porritt Social Assay London 1962

68. 6th Nov 1967 Kenneth Robinson Statement to HC
HC Debates Vol 753

69. Ministry of Health
The Administrative Structure of the Medical And Related Services in England and Wales
HMSO London 1968

70. Department of Health & Social Security
The Future Structure of the National Health Service
HMSO London 1970

71. Department of Health & Social Security
Reform of Local Government in England
HMSO London 1970 Cmnd 4276

72. Department of Health & Social Security
National Health Service Reorganisation: England
HMSO London 1972 Cmnd 5055

73. Department of Health & Social Security
The National Health Service Re-Organisation Act 1973
HMSO London 1973 Eliz II Ch 32

74. Department of Health & Social Security
Management Arrangement for the Re-organised Health Service
HMSO London 1972

75. Department of Health & Social Security
Report Of The Royal Commission on the National Health Service.
Royal Commission Chair: Merrison A
Cmnd 7615 HMSO London July 1979

76. Perrin J et al
Management of Financial Resources in the National Health Service
Royal Commission On The National Health Services, Research Paper 2, HMSO London 1978

77. Department of Health & Social Security
Sharing Resources for Health in England (RAWP)
HMSO London 1976

78. Department of Health & Social Security and The Welsh Office
Patients First
HMSO London 1979

79. Department of Health & Social Security
Health Services Development: Structure and Management
HC(80)8 July 1980

80. Department of Health & Social Security
Management Services Information Requirements Of The Health Services
HN(79)21 February 1979

81. Department of Health & Social Security
First report to the Secretary of State
Steering Group on Health Services Information
Chair: Mrs Edit Korner HMSO London 1982

82. Department of Health & Social Security
Health Services Development: The NHS Planning System
HC(82)6 January 1982

83. Department of Health & Social Security
Health Services Development: Annual Regional Review Meetings
HN(82)3 January 1982

84. Department of Health & Social Security
Health Services Management: Performance Indicators
HN(83)25 August 1983

85. Department of Health & Social Security
Health Services Management: Performance Indicators
HC(85)23 September 1985

86. Department of Health & Social Security
Health Services Management: Performance Indicators
HN(85)32 November 1985

87. Department of Health & Social Security
Health Support Services: Contracting Out
HC(83)14 April 1983

88. Department of Health & Social Security
Competitive Tendering In The Provision of Domestic, Catering and Laundry Services
HC(83)18 May 1983

89. Department of Health & Social Security
Competitive Tendering In The Provision of Domestic and Laundry Services (Specimen Contracts)
DA(83)40 July 1983

90. Griffiths R
NHS Management Enquiry
Letter to Rt Hon Norman Fowler MP 6th October 1983

91. Department of Health & Social Security
Health Services Management Implementation of the NHS Management Inquiry Report
HC(84)13 June 1984

92. Knox G
Health care Information
Nuffield Provincial Hospitals Trust Occasional Paper 8

93. Department of Health & Social Security
Resource Management (Management Budgeting) in Health Authorities
HN(86)34 1986

94. Brunel University Health Economics Research Unit
Resource management, Process and Progress. Monitoring the Six Acute Hospital Sites. Interim Report of the Brunel University Evaluation Team
Brunel University Health Economics Research Group 1989
Buxton M, Packwood T, Keen J

95. Department of Health & Social Security
Working For Patients
HMSO London 1988

96. Department of Health and Social Security
Working For Patients: Framework For Information Systems: The Next Steps
HMSO London 1990

97. Kelly et al vs Dawes
The Times Newspaper Law Reports
July 14th 1989

98. Burke vs London Borough of Tower Hamlets
The Times Newspaper Law Reports
August 10th 1989

100. Cook S
Modelling Generic User Interfaces With Functional Programs
In: Harrison MD, Monk AF (eds)
People & Computers: Designing For Usability. Proceedings of the Second Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1986 pp369-385

101. Moris A
Expert Systems -Interface Insight
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the Third Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp307-324

102. Berry D, Broadbent R
Expert Systems & The Man Machine Interface: Pt2: The User Interface
Expert Systems Feb 1987 Vol 4 No 1 pp56-68

103. Checkland PB
Information Systems and Systems Thinking: A Time To Unite
International Journal Of Information Management
1988 No 8 pp239-248

104. Bright L, Inman A, Stammers R
Human Factors In Expert Systems Design: Can Lessons In The Promotion Of Methods Be Learned From Commercial DP?
Interacting With Computers Aug 1989 Vol 1 No 2 pp141-151

105. Pejtersen AM, Rasmussen J
Design & Evaluation Of User-System Interfaces or of User-Task Interaction: A Discussion of Interface Design Approach In Different Domains For Application of Modern IT
In: Zunde P, Agraval C (Eds)
Empirical Foundations Of Information and Software Science IV: Empirical Methods of Man Machine Interfaces
North Holland 1988 pp33-62

106. Wilson MD, Barnard PJ, Green TR, Maclean A
Knowledge Based Task Analysis for Human Computer Systems
In: Van Der Veer, Green G, Hoc T, Murray DM (Eds)
Working With Computers: Theory and Outcome
London Academic Press 1988 pp47-87

107. Alexander H
Formally Based Techniques for Dialogue Design
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the 3rd Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp201-213

108. Friis S
Action Research On Systems Development: Case Study Of Changing Actor Roles
Computers & Society January 1988 Vol 18 No 1 pp22-31

109. Rushinek A, Rushinek SF
End User Satisfaction Of DBMS
Database Winter 1986 Vol 17 No 2 pp17-25

110. Dillon A
Knowledge Acquisition & Conceptual Models: A Cognitive Analysis Of The Interface
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the Third BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp371-379

111. Pliskin N, Shoval P
End User Prototyping: Sophisticated Users Supporting System Development
Database Sept 1987 Vol 18 No 4 pp7-12

112. Reynolds CF
Human Factors in Systems Design: A Case Study
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the Third Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp93-102

113. Norman D
Cognitive Engineering
In: Norman D, Draper S (eds)
User Centred System Design
Lawrence Erlbaum Associates 1986 p61

114. Bjφrn-Andersen N
Are Human Factors Human?
The Computer Journal 1988 Vol 31, No 5 pp386-390

115. Land F
The Impact of Information Technology On The Workplace
In: Piercy N
The Management Implications of New Technology
Croom Helm Publishing 1984 pp75-94

116. Laurel B
Interfaces As Mimesis
In: Norman D, Draper S (eds)
User Centred Systems Design
Lawrence Erlbaum Associates 1986

117. Green TRG
Limited Theories As A Framework For Human-Computer Interaction
In: Ackerman D, Tauber M (eds)
Mental Models in Human Computer Interaction
North Holland 1990

118. Polson P, Lewis C
Theory Based Design For Easily Learned Interfaces
Human Computer Interaction 1990 Vol 5 No 2 pp191-220

119. Card SK, Moran TP, Newell A
The Psychology of Human Computer Interaction
Hullsdale NJ Lawrence Erlbaum Associates 1983

120. Dillon A
Knowledge Acquisition & Conceptual Models: A Cognitive Analysis of the Interface
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the Third Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp371-379

121. Carey MS, Stammers RB, Astley JA
Human Computer Interaction Design: the Potential and Pitfalls of Hierarchial Task Analysis
In: Diaper D (Ed)
Task Analysis for Human Computer Interaction
Elis Horwood 1989 pp56-63

122. Olson JR, Olson G
The Growth Of Cognitive Modelling In Human Computer Interaction Since GOMS
Human Computer Interaction 1990 Vol 5 No 2 pp221-265

123. Brocklehurst ER
The NPL Electronic Paper Project
International Journal Of Man Machine Studies
1991 Vol 34 pp69-95

124. Bjϕn-Anderson
Understanding The Nature Of The Office For The Design of Third Wave Systems
In: Harrison MD, Monk AF (eds)
People & Computers: Designing For Usability. Proceedings of the Second Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1986 pp65-77

125. Karat J, McDonald E, Anderson M
A Comparison Of Menu Selection Techniques: Touch Panel, Mouse and Keyboard
International Journal Of Man Machine Studies
1986 Vol 25 pp73-88

126. Dillon , McKnight, Richardson
Reading From Paper vs Reading From Screen
Computer Journal Oct 1988 Vol 31 No 5 pp457-464

127. Haas C
Does The Medium Make A Difference? Two Studies of Writing With Pen and Paper and With Computer.
Human Computer Interaction 1989 Vol 4 pp149-170

128. Lamberli D, Wallace WA
Presenting Uncertainty in Expert Systems: An Issue in Information Portrayal
Information & Management 1987 Vol 13 pp159-169

129. Morris A
Expert Systems - Interface Insight
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the Third Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp307-324

130. Kidd AL, Cooper MB
Man Machine Interface Issues in Const and Use of an Expert System
International Journal Of Man Machine Studies
1985 Vol 22 pp91-102

131. Codd EF
How About Recently?
In: Shneiderman B (ed)
Proceedings International Conference on Databases: Improving Usability & Responsiveness
Academic Press 1978 pp3-12

132. Blanning RW
Conversing With MIS In Natural Language
Communications of ACM 1984 Vol 27 No 3 pp201-207

133. Hill E
What We Do Not Know About Man-Machine Systems
International Journal Of Man-Machine Studies
1986 Vol 18 pp135-143

134. Ringle M, Halslead-Nussloch R
Shaping Users Input: A Strategy for Natural Language Dialogue Design
Interacting With Computers Dec 1989 Vol 1 No 3 pp227-244

135. IBM Corp & Microsoft Inc
The Programming Family (DOS Version 3.3)

136. International Computers Limited
QueryMaster: Using QueryMaster
ICL Documentation: ROO433/02

137. Hayes PJ & Reddy PA
Graceful Interaction Through The Cousin Command Interface
International Journal Of Man Machine Studies
1983 Vol 19 No 3 pp231-284

138. Carey T
User Differences In Interface Design
Computing November 1982 pp14-23

139. Zloof MM
QBE/QBO: A Language For Office and Business Administration
IEEE Computer May 1981 Vol 14 No 5 pp13-22

140. Gittins D
Icon Based Human Computer Interaction
International Journal of Man Machine Studies
1986 Vol 24 pp519-543

141. Fairchild K, Mereditch G, Wexelbat A
A Formal Structure for Automatic Icons
Interacting With Computers Aug 1989 Vol 1 No 2 pp131-140

142. Took R
Text Representation & Manipulation In A Mouse Driven Interface
In: Harrison MD, Monk AF (eds)
People & Computers: Designing For Usability. Proceedings of the Second Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1986 pp386-402.

143. Hitch G, Sutcliffe G, Bowers J, Eccles L
Empirical Evaluation OF Map Interfaces: A Preliminary Study
In: Harrison MD, Monk AF (eds)
People & Computers: Designing For Usability. Proceedings of the Second Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1986 pp 98-104

144. Kilgour A
Theory & Practice in User Interface Management Systems
Systems Journal May 1987 Vol 29 No 4 pp171-174

145. Pfaff GE (ed)
User Interface Management Systems
Springer-Verlag, Berlin West Germany, 1985

146. Cockton G
Interaction Ergonomics, Control and Separation: Open Problems in User Interface Management
Information & Software Technology 1987 Vol 29 No 4
pp176-191

147. Newman W
An Experimental Program for Architectural Design
Computer Journal 1966 Vol 9 No 1 pp21-26

148. Cockton G
Generative State Transition Networks: A New Communication Control Abstraction
Scottish HCI Centre Report AMU8830/01H Mar 1988

149. Gray p, Kilgour A, Wood A
Dynamic Reconfiguability for Fast Prototyping of User Interfaces
Software Engineering Journal Nov 1988 pp257-262

150. Kasik DJ
Controlling User Interaction
Computer Graphics 1976 Vol 10 No 2 pp91-106

151. Barker P, Manji K
Pictorial Knowledge Bases
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the Third Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp162-173

152. Hammond N, Allinson L
The Travel Metaphor A Design Principle & Training Aid For Navigating Around Complex Systems
In: Diaper D, Winder R
People & Computers III: Proceedings of BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp75-89

153. Nielsen J
Hypertext and Hypermedia
Academic Press 1990

154. Monk A
The Personal Browser: A Tool For Directed Navigation In Hypertext Systems
Interacting With Computers Aug 1989 Vol 1 No 2 pp190-196

155. Weyer S, Borning A
A Prototype Electronic Encyclopedia
ACM Transactions On Office Systems Vol 3 pp63-68

156. Weyer SA
Searching For Information In A Dynamic Book
Palo Alto Research Centre 1982 Report No SCG-82-1

157. Brown A, Took R
Design and Construction of Graphical Database User Interfaces Using Surface Interaction
In: Brown A, Hitchcock P (eds)
Proceedings of the Eighth British National Conference on Databases
Cambridge Press 1990 pp243-262

158. Kindborg M, Kollerbaun A
Visual Languages & HCI
In: Diaper D, Winder R (eds)
People & Computers III: Proceedings of the Third Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1987 pp175-187

159. Schofield J
Rooms For Manoeuvre
The Guardian Computer Review 18th June 1991

160. Sutcliffe AG
Use Of Conceptual Maps As Human Computer Interfaces
In: Johnson P, Cook S
People & Computers: Designing The User Interface. Proceedings of the Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1985 pp117-128

161. Hitch G, Sutcliffe G, Bowers J, Eccles L
Empirical Evaluation of Map Interfaces: A Preliminary Study
In: Harrison MD, Monk AF (eds)
People & Computers: Designing For Usability. Proceedings of the Second Conference of the BCS Human Computer Interaction Specialist Group
Cambridge Press 1986 pp565-585

162. Truckenbrod JR
Effective Use of Color in Computer Graphics
ACM SIGGRAPH Conference Proceedings
1981 Vol 15 No 3 pp83-90

163. Webb JB Kramer A
Maps or Analogies: A Comparison of Instructional Aids for Menu Navigation
Human Factors June 1990 Vol 32 No 3 pp251-266

164. Nielsen J
The Matters that Really Mater for Hypertext Usability
Proceedings Hypertext 1989 Association of Computer Machinery New York pp239-248

165. Nielsen J
The Art Of Navigating Through Hypertext
Communications of the ACM Mar 1990 Vol 33 No 3 pp296-310

166. Card S Polson P
Human Computer Interaction General Intro Notes
Human Computer Interaction 1990 Vol 5 No 2 pp119-123

167. Shneiderman Ben
Designing The User Interface
Addison-Wesley May 1987

168. Robertson D
Helping Inexperienced Users To Construct Simulation Programs: An Overview Of The ECO project
In: Bramer MA
Research & Development In Expert Systems IV
1988 British Informatics Society Ltd pp185-197

169. Rich E
Users Are Individuals: Individualizing User Models
International Journal Of Man Machine Studies
1983 Vol 18 pp199-214

170. Innocent PR
Towards Self Adaptive Interface Systems
International Journal Of Man Machine Studies
1982 Vol 16 pp287-299

171. Hockley AT
Adaptive User Interfaces For Information Systems: An Evaluation
In: Zunde p, Agrawal JC (Eds)
Conference Proceedings: Empirical Foundations of Information & Software Science IV In: Empirical Methods of Evaluation of Man Machine Interfaces
North Holland 1988 pp149-162

172. Benyon, Murray
Experiences With Adaptive Interfaces
The Computer Journal Oct 1988 Vol 31 No 5 pp465-473

173. Liang T
User Interface Design for DSS: A Self Adaptive Approach
Information & Management 1987 Vol 12 pp181-193

174. MacLean A, Carter K, Lovstrand L, Moran T
User-Tailorable Systems: Pressing the Issues With Buttons
In: Proceedings CHI'90 Seattle Washington April 1990

175. Sandstrom G
Implementing Modifications To Information Systems While Using Them: An Intentionally Intervening Approach
In: Zunde P, Agrawal J (Eds)
Empirical Foundations of Information & Software Science IV Empirical Methods of Evaluation of Man Machine Interfaces
North Holland 1988 pp273-286

176. Innocent PR
Towards Self Adaptive Interfaces
International Journal Of Man Machine Studies
1982 Vol 16 pp287-299

177. Panko RR & Sprague RH
Implementing Office Systems Requires A New DP Outlook
Data Management November 1984 pp40-42

178. Date CJ
An Introduction To Database Systems
Addison-Wesley 1981

179. Green S Devlin S Cannata P, Gomez L
No IF's, AND's or OR's: A Study of Database Querying
International Journal Of Man Machine Studies
1990 Vol 32 pp303-326

180. Oxborrow EA
Databases & Database Systems: Concepts & Issues
Chartwell Bratt 1989

181. Jardine DA (ed)
The ANSI/SPARC DBMS Model
North Holland 1977

182. Chen PP
The Entity Relationship Model: Towards A Unified View of Data
ACM Transactions On Database Systems
March 1976 Vol 1 No 1 pp131-142

183. Bodie ML
On the Development of Data Models
In: Schmidt JW
On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases and Programming Languages. Springer Verlag 1984 pp19-47

184. Tsichritzis D, Lochovsky F
Data Models
Prentice Hall Inc 1982

185. Daly WG
A Survey Of Graphical Interfaces for Database Management
University of York Internal Report YCS 135 1990

186. Committee for Advanced DBMS Function
Third Generation Database System Manifesto
In: Proceedings of the IFIP TC2 Conference on Object Oriented Databases 1989

187. CODASYL
CODASYL Data Description Language Committee Journal Of Development
Canadian Gov Publishing Centre 1981

188. Date CJ
An Introduction to Database Systems
Addison Wesley 1986

189. Zloof MM
Query By Example: A Database Language
IBM Systems Journal 1977 Vol 16 No 4 pp324-343.

190. McDonald N, Stonebraker M
CUPID: The Friendly Query Language
In: Proceedings of ACM-Pacific April 1975 pp127-131

191. Stonebaker M, Kalash J
TIMBRE: A Sophisticated Relation Browser
In: Proceedings of the Eighth International Conference on Very Large Databases 1982 pp1-10

192. Hendersen DA, Card SK
Rooms: The Use of Multiple Virtual Workspaces to Reduce Space Contention in a Window-Based Graphical User Interface
ACM Transactions on Graphics July 1986 Vol 5 No 3
pp 211-243

193. Chan EPF, Lochovsky FH
A Graphical DataBase Design Aid Using the Entity Relationship Model
In: Entity-Relationship Approach to Systems analysis and Design
North Holland 1979 pp295-303

194. Bragger R et al
GAMBIT: An Interactive Database Design Tool for Data Structures, Integrity Constraints and Transactions
In: Proceedings of the IEEE Conference on Data Engineering Los Angeles 1984 pp399-407

195. Wong HKT, Kuo I
GUIDE: Graphical User interface for Database Exploration
In: Proceedings of the Eighth Conference on Very Large Databases
Mexico City 1982 pp22-31

196. King R
A Database Management System Based on an Object Oriented Model
In: Kerschberg L (ed)
Expert Database Systems: Proceedings from the First International Workshop
The Benjamin/Cummins Publishing Co Inc 1986 pp443-467

197. Goldman SA, Kanellakis P, Goldman S, Zdonik S
ISIS: Interface for a Semantic Information System
In: Proceedings of ACM SIGMOD International Conference Management of Data
1985 pp328-342

198. Hammer M, McLeod D
Database Description with SDM: A Semantic Database Model
ACM Transactions on Database Systems 1981 Vol 6 No 3
pp351-386

199. Bryce D, Hull R
SNAP: A Graphics Based Schema Manager
In: Proceedings of IEEE Conference on Data Engineering
Los Angeles California 1986

200. Abiteboul S & Hull r
IFO: A Formal Semantic Database Model
ACM Transactions on Database Systems December 1987 Vol 12 No 4 pp525-565

201. Santucci G, Sottile PA
Query By Diagram: A Visual Environment for Querying Databases
(To appear IEEE Systems Journal)

202. Roberts SA, Gahegan MN
Supporting the Notion of Context Within a Database Environment for Intelligent Reporting and Query Optimisation
European Journal of Information Systems
1991 Vol 1 No 1 pp13-22

203. Maier D, Nordquist P, Grossman M
Displaying Database Objects
In: Kerschberg L (ed)
Expert Database Systems
The Benjamin/Cummings Publishing Co 1987

204. Young L
A Systems Architecture for Supporting Senior Managers Messy Tasks
Information & Management 1987 Vol 13 pp85-94

205. Teorey TJ, Yang D, Fry JP
A Logical Design Methodology for Relational Databases Using the Extended Entity-Relational Model
ACM Computing Surveys June 1986 Vol 18 No 2 pp197-221

206. Codd EF
Extending The Database Relational Model to Capture More Meaning
ACM Transactions On Database Systems
December 1979 Vol 4 No 4 pp397-434

207. Lazarevic B, Misic V
Extending The ER Model To Capture Dynamic Behaviour
European Journal Of Information Systems 1991 Vol 1 No 2 pp95-106

208. Goldschmidt D, Reiter A
An Implementation Of FORAL Using IDMS
In: Shneiderman B (ed)
Proceedings International Conference On Databases: Improving Usability and Responsiveness
Academic Press 1978 pp131-143

209. Zhaq L, Roberts SA
An Object Orientated Data Model For Database Modelling, Implementation & Access
BCS Journal 1988 Vol 13 No 2 pp117-125

210. Conklin J
Hypertext: An Introduction and Survey.
IEEE Computer Sept 1987 Vol 20 No 9 pp17-41

211. Meyer B
Object Oriented Software Construction
Prentice Hall 1988

212. Pinson LJ & Wiener RS
An Introduction to Object Oriented Programming and Smalltalk
Addison-Wesley 1988

213. Krasner G
Smalltalk-80: Bits of History, Words of Advice
Addison-Wesley 1983

214. Deutsch LP
The Past, Present, and Future of Smalltalk
European Conference on Object Orientated Programming
1989 Springer Valeg pp73-87

215. Weyer SA
The Design of a Dynamic Book for Information Search
International Journal For Man Machine Studies July 1982 Vol 17 No 1 pp87-107

216. Knolle N
Variations of Model-View-Controller
Journal of Object Oriented Programming September/October 1989 pp42-46

217. Parc Place Systems
Mountain View
California US

218. Blake E, Cook S
On Including Part Hierarchies in Object Oriented Languages, with an Implementation in Smalltalk
Proceedings of the European Conference on Object Oriented Programming June 1987 pp41-50 Springer Valeg

219. Prospero Software Ltd London
Prospero Pascal For GEM
August 1987

220. Digital Research
GEM Programmers ToolBox
1987

221. Howling B & Pepper A
A Programmers Guide To GEM
SIGMA Press Ltd 1988

222. Goldberg A & Robson D
Smalltalk-80: The Language and its Implementation
Addison-Wesley 1983

223. Nye A (ed)
XLIB Reference Manual Version 11
O'Reilly Associates July 1990

224. Young D
Window Systems Programming & Applications With X
Sigma Press 1989

225. Scheifier R, Gettys J
The X Window System
ACM Transactions on Graphics Vol 5 Nr 2 April 1986 pp79-109

226. Cook S
Object Oriented Programming & its Applications
In: Object Oriented Programming
Strategies for Interface and Display Management and Other Applications
UNICOM / BCS Object Oriented Programming Systems Group 1988 pp2-13

227. Hopkins T
Smalltalk-80: An Object Oriented Programming Tool
EuroGraphics UK 7th Conference March 1989 Tutorial II

228. Goldberg A
Introducing the SmallTalk-80 System
Byte August 1981 Vol 6 pp14-26

229. Linton M, Vlissides JM, Calder PR
Composing User Interfaces With Interviews
IEEE Computer Feb 1989 No 2 Vol 22 pp8-22

230. Booch G
Object Oriented Software Design With Applications
Ch 17 Object Oriented Programming in Classical Languages pp375-383 & Ch 20 pp440-443

231. Ingalls DHH
The Evolution of the Smalltalk Virtual Machine
In: Krasner G (ed)
Smalltalk-80: Bits of History, Words of Advice
Addison Wesley 1983

232. LaLonde W & Pugh J
Designing Is Hard: Object Oriented Software Is Different
Journal Of Object Oriented Programming March/April 1989 pp47-55

233. Pratt TW
Programming Languages: Description & Implementation
Prentice Hall 1984

234. Shneiderman B
Software Psychology: Human Factors and Information Systems
Cambridge Mass: Winthrop Press 1980

235. Krasner G & Pope S
A CookBook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80
Journal of Object Oriented Programming August/September 1988 pp26-49

236. Adams S
MetaMethods: The MVC Paradigm
HOOPLA July 1988 pp5-21

237. LaLonde W & Pugh J
Pluggable Tiling Windows
Journal of Object Oriented programming August/September 1989 pp57-66

238. Burns L, Archbald J, Malhtora A
A Graphical Entity-Relationship Database Browser
In: Shriver B (ed)
Software Track: Proceedings of the Twenty First Annual International Conference
IEEE Computer Society 1988 pp694-704

239. Zhaq L, Roberts SA
An Object Oriented DataModel for DataBase Modelling, Implementation and Access.
BCS Journal Vol 31 No 2 pp117-121 1988

240. Norman DA
Design Principles for Human Computer Interfaces
In: Janda (ed)
Proceedings of CHI'83 Conference on Human Factors in Computer Systems Addison Wesley pp41-48

241. Stocker & Cantie
A Target Logical Schema: The ACS
Internal Report CSA/5/1983 School Of Computing & Accounting UEA 1983

242. Robb FF
Prescription for the Support of Groups Addressing Novel Organisational Problems
International Journal Of Information Management 1988 Vol 8 pp275-288

APPENDIX ONE

Background and Interrelationship of PAS and DIS

PAS was introduced to solve several practical management problems:

- the patient index was usually located in one place, the medical records department. All other departments had to physically go there and borrow records as required. This led to bottle necks and delays especially if several departments requested the same patient details at the same time.

- Often a patient's records are stored in another location, or worse if scattered over several functions or maybe even another hospital. This does now allow for effective updating of information and takes time to transfer to the requesting department.

- Cards are handwritten, often under great pressure leading to mistakes and problems of handwriting illegibility.

- Clerical mistakes occur - cards are misfiled, and transcription errors arise as detailed information is repeatedly copied.

- Several copies of the same information are maintained within the system, leading to problems of inconsistency when one is updated.

- Departments compiled and managed their own information indexing systems, so the total information about a patient may be scattered throughout the hospital or at worst the District.

- Public expectation changed: no longer did the public wish to repeatedly give the same information to what was perceived as a single Service, rather the information should be given just once. Also the public began to expect better organisation and management, for example of outpatient clinics it was expected that the clinic would be ready to receive them at a ceratin time, rather than having to wait excessive periods of time.

- In addition information for planning and management was needed from this information resource at both national, Regional, District and unit level.

The Region in the study standardised on the ICL PAS product. Other Regions have taken products from a multitude of suppliers, however the basic components of the system remain the same. The ICL PAS was supplied in a series of modules, designed to be mounted side by side on the same hardware processor. Modules could be taken on in any order at any time, except for the first base module which was mandatory.

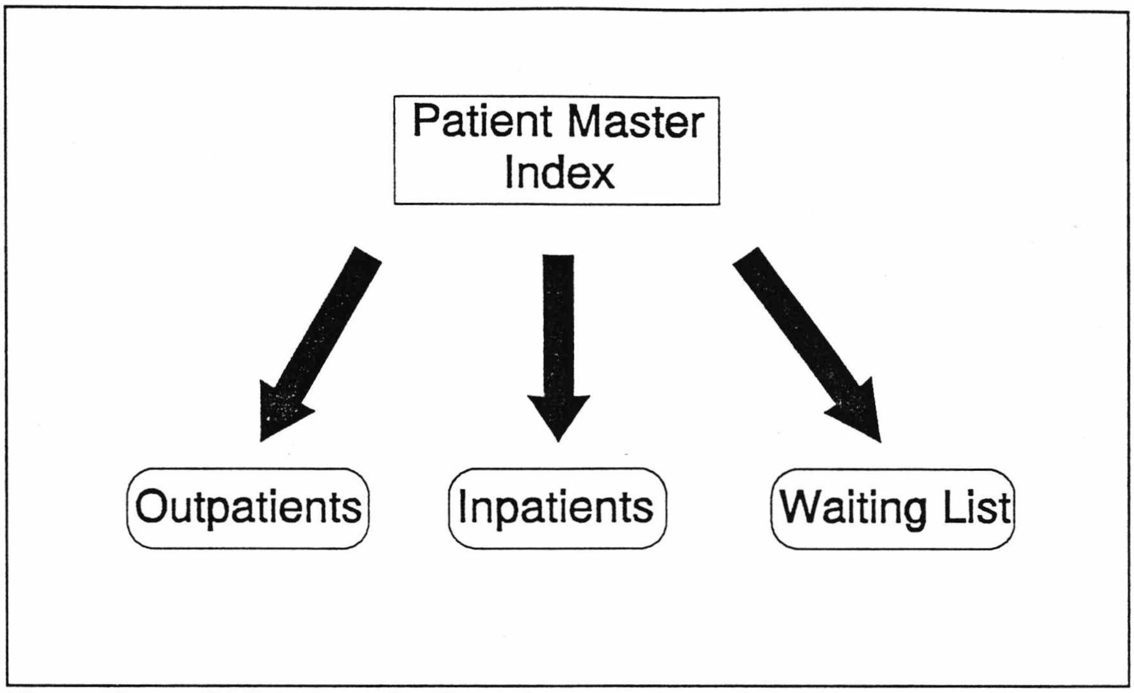


Figure 1 PAS Internal Configuration

As presented in Figure Fifty, the modules comprising PAS are:

Patient Master Index(PMI)

This module contains the patient master index and the health care contacts register. The PMI holds record for each patient containing personal information such as name, date of birth, sex, and GP. The Health Care Contacts Register is made up of the Hospital Contacts Register and the GP register. Each of these subsidiary registers hold personal information relating the either the consultant or the GP as required.

The PMI forms the hub of PAS, for this reason it is mandatory and always the first module to be implemented.

Inpatients Module

This module holds administrative information arising from a patients stay in a hospital, for example wards stayed on. This forms the central component for the routine management of the hospital. Standard reports are available giving bed and ward state information. Throughput statistics are also available forming the basis for HAA returns.

Outpatients Module

This module manages the outpatient clinic functions. In particular appointment information is managed, providing a booking management system. Clinics can now be effectively rearranged and letters produced automatically. Each clinic is built up around a set of rules, for example two patients every hour then one an hour in the afternoon.

Waiting List Module

Prior to this module each consultant maintained their own waiting list in their own fashion. This provided little reliable information for management and consultants alike, for example as many patients were entered on several lists. This module provides a single non disputable source of managed waiting list information.

District Information Service

Alongside PAS is mounted DIS. DIS provides a historical activity profile of the District, being fed from PAS via daily tape updates. PAS and DIS are mounted on the same processor but accessed by different user groups. For example admissions staff admit patients onto PAS producing IPREC forms at the admissions desk, medical records staff enter the final diagnostic codes and Information Management produce ad hoc management reports from DIS.

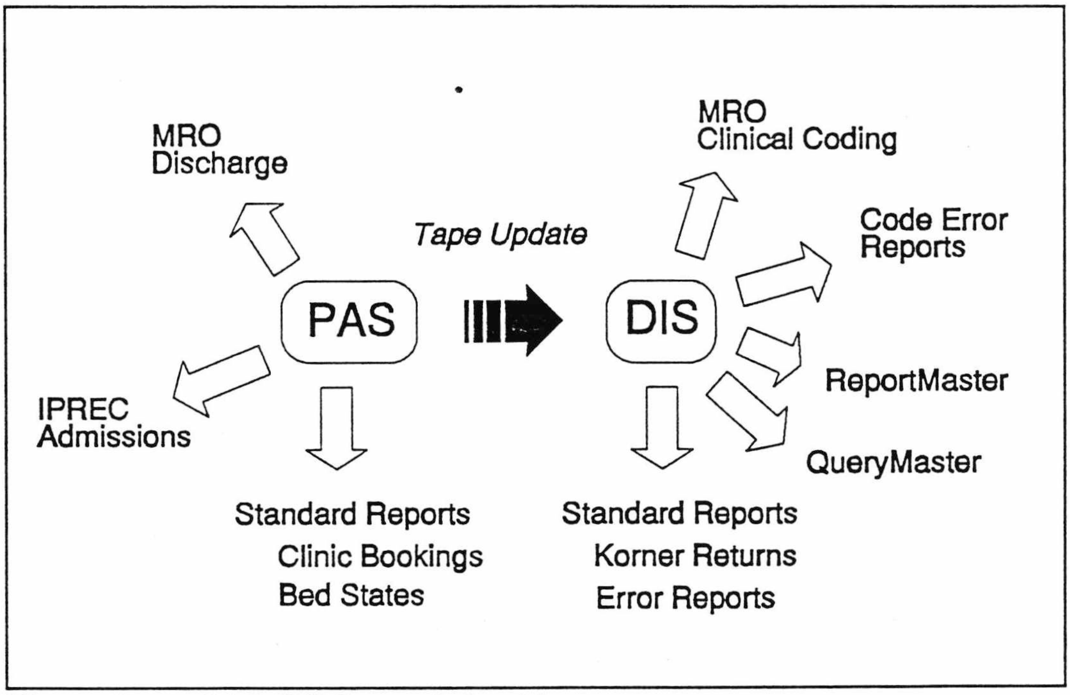


Figure 2 PAS DIS Inter-relationship

DIS is an IDMS ICL database product produced in collaboration with the IRC. There are three key components to the data model

- ward stay — This relates to all activity between the patient and the hospital during a single period on a single ward. If a patient is transferred to another ward or hospital a new ward stay is started.

- consultant episode This relates to all activity between the patient and the hospital during a single period with a single consultant. Just as in transferring wards, if the patient is transferred to another consultant, another consultant episode is started.
- District Spell This relates to all activity between the patient and the District during a patients stay within the District. Just as with ward stay and consultant episode, if the patient is transferred to another District then a new District Spell is started.

The aim of this approach to modelling patient interaction has been to provide a detailed account of patient activity from a Regional rather than District perspective. The datamodel was developed towards the goal of Regional information integration with each District providing feeder information.

APPENDIX TWO

District Profiles

The aim of this appendix is to provide a more detailed profile of the Region as a whole and for each individual District which participated in the study. This is presented primarily in diagrammatic form covering three key components:

- population characteristics
- resourcing characteristics
- organisations structures employed

Population Characteristics

One of the key determinants of health care demand is population size and structure. In general terms an increase in the size and age of the population results in increased demand for health care. The total population of the Region has stayed at approximately 3.6 million since 1978 but has experienced a significant change in structure. The Region is the sixth most populous in England but has one of the largest proportions of elderly people. Between 1978 and 1983 the under 15 age group has declined in all Districts within the Region, whereas the 75 plus age group has increased by 12% across the Region as a whole.

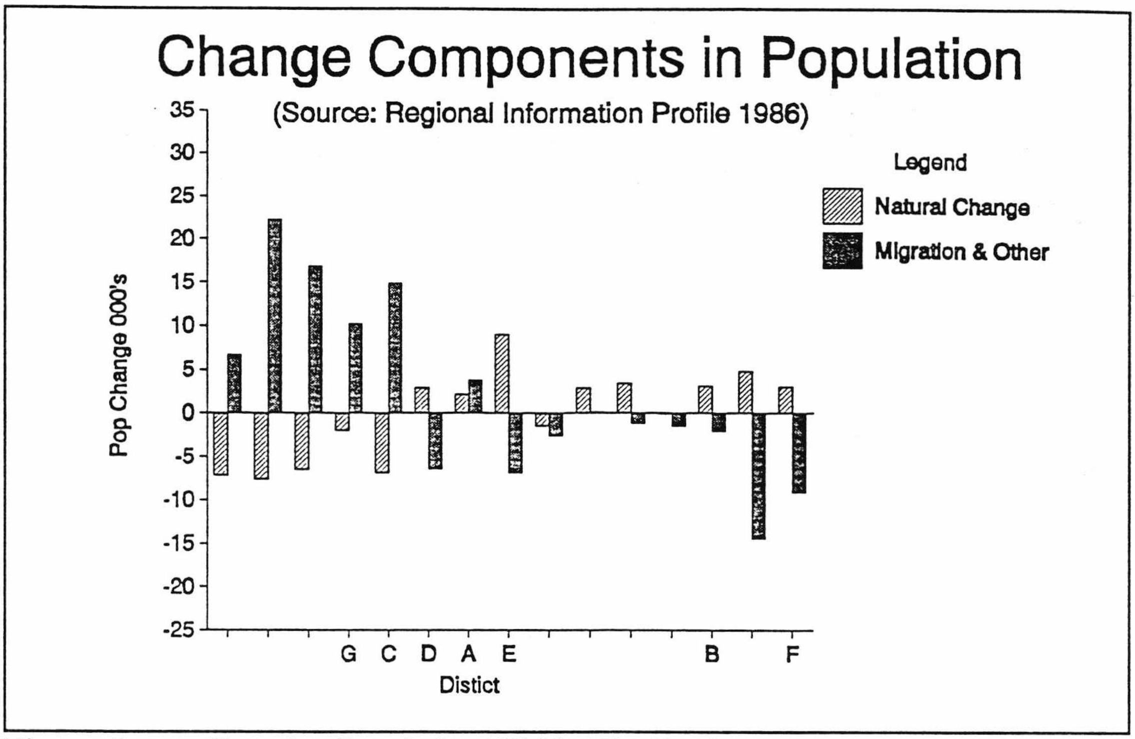


Figure 3 Changing Population Within The Region

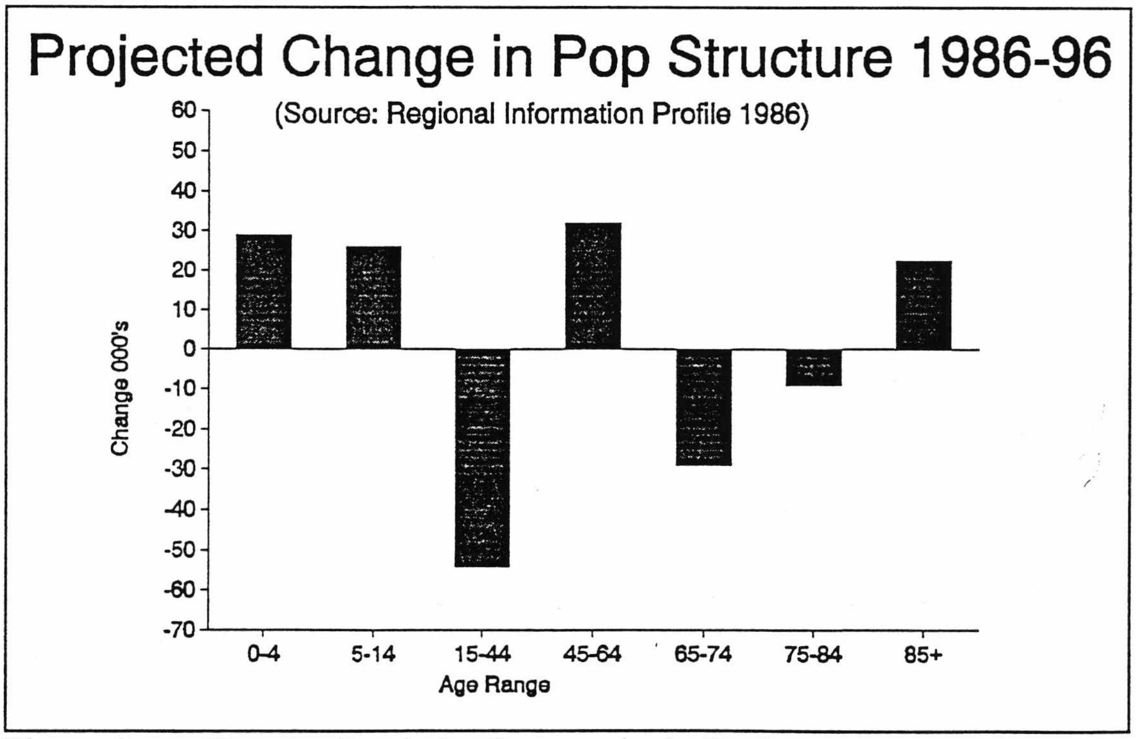


Figure 4 Changing Population Structure in the Region

Resourcing

One of the principle constraints on health care provision is that of resourcing. Since 1976 a standardised resource allocation has been applied across Districts tieing resource allocation to its demographic population profile. In 1990 this formulae was abandoned in favour of a resourcing per capita of resident population. A summary of the Regional allocation is given in Figure Fifty Four.

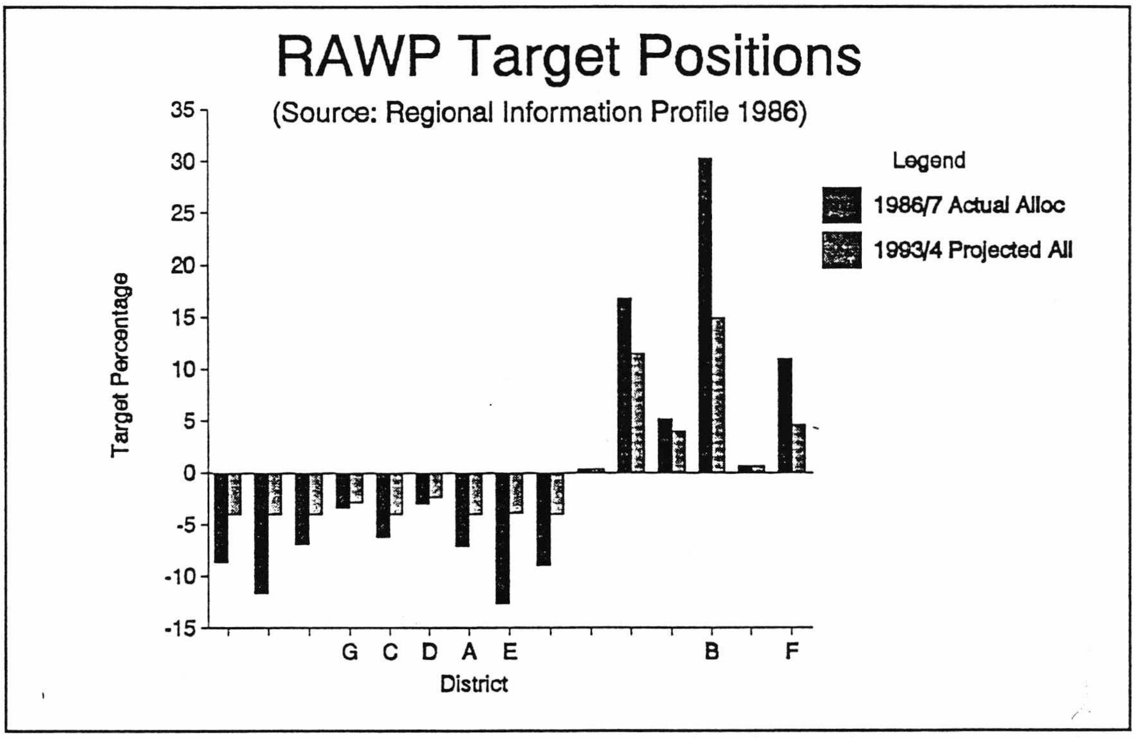


Figure 5 RAWP Formulae Target and Allocation Comparison

A key component of resourcing for health care is capital allocations made by each District. Each District is free to allocate capital investment as required within guidelines established as part of the five year planning process. As a whole the Region spent 12.3% (£9.4 million) on capital hard and software expenditure in 1986/7. This has been budgeted to remain at this level of commitment until the end of the current strategic planning period in 1994.

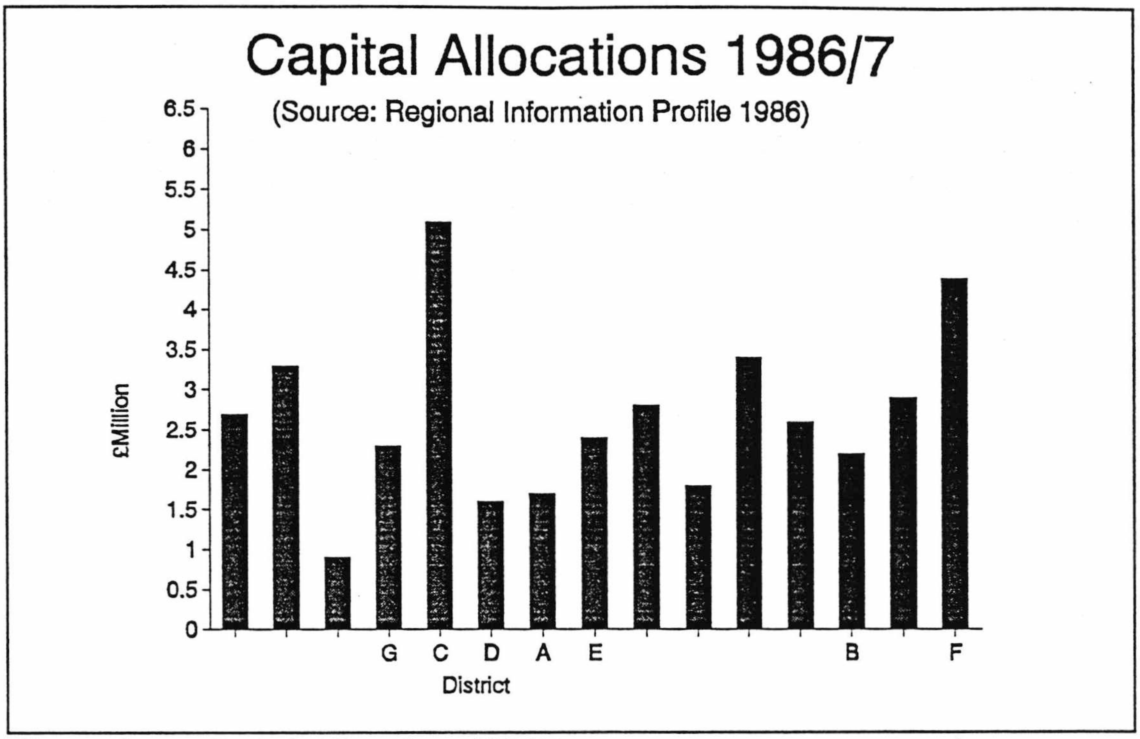


Figure 6 Capital Allocations 1986/7

Local Level Activity and Organisation

The Region provided 9062 bed days in 1986/7, a fall of 11.5% since 1982. One proxy measurement for the utilisation of this resource is activity, that is death and discharge rates as a measure of patient throughput. This is summarised in Figure Fifty Six showing throughput per District. Despite the fall in bed numbers, discharge rates have increased by 5.2% over the same period.

Each District has been able to develop its own organisational structures to maintain this delivery of health care. Concentrating on the information function exclusively, this can be seen to vary considerably between Districts. The remainder of this Appendix presents detailed organisation charts for each District visited in the study carried out in Chapter Three.

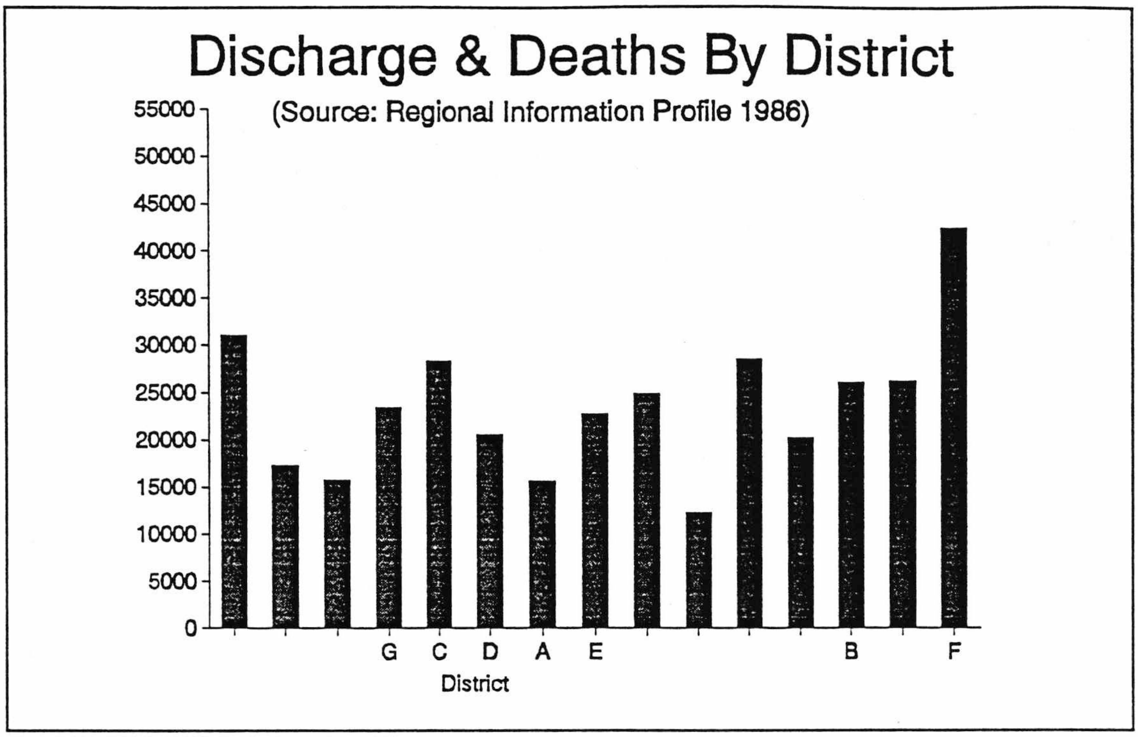
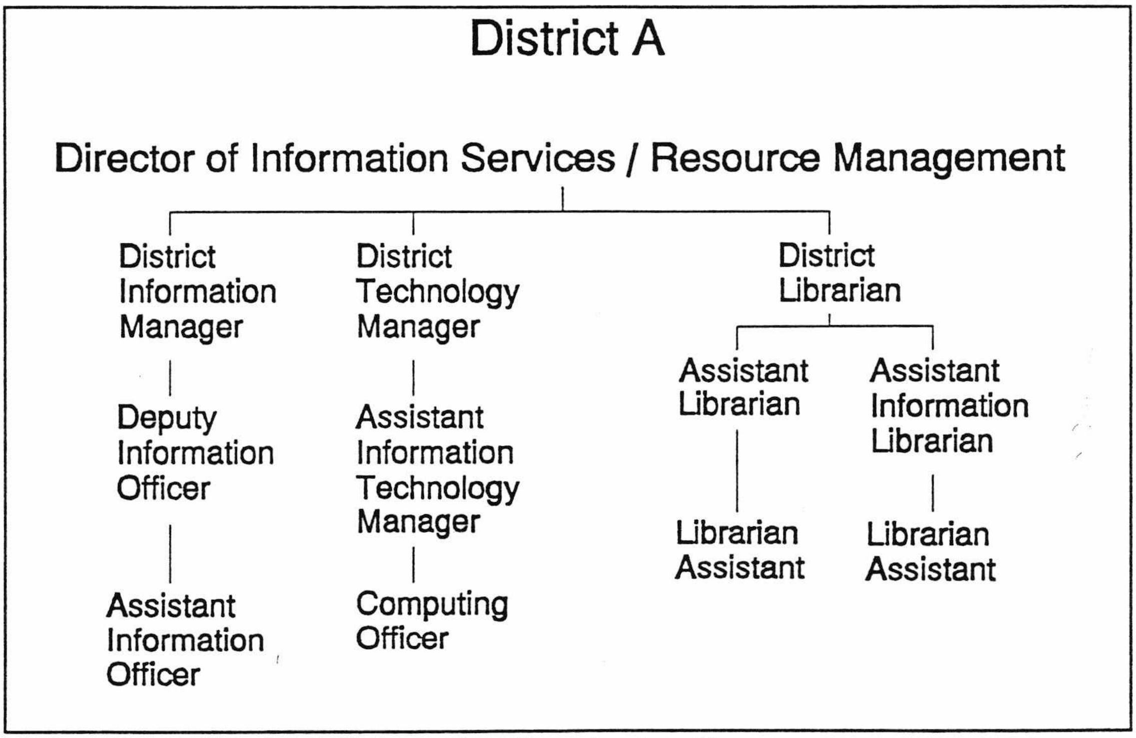


Figure 7 Throughput Rates Per District



District A

Director of Information Services / Resource Management

District Information Manager
Deputy Information Officer
Assistant Information Officer

District Technology Manager
Assistant Information Technology Manager
Computing Officer

District Librarian
Assistant Librarian
Librarian Assistant
Assistant Information Librarian
Librarian Assistant

Figure 8 District A Information Organisation Chart

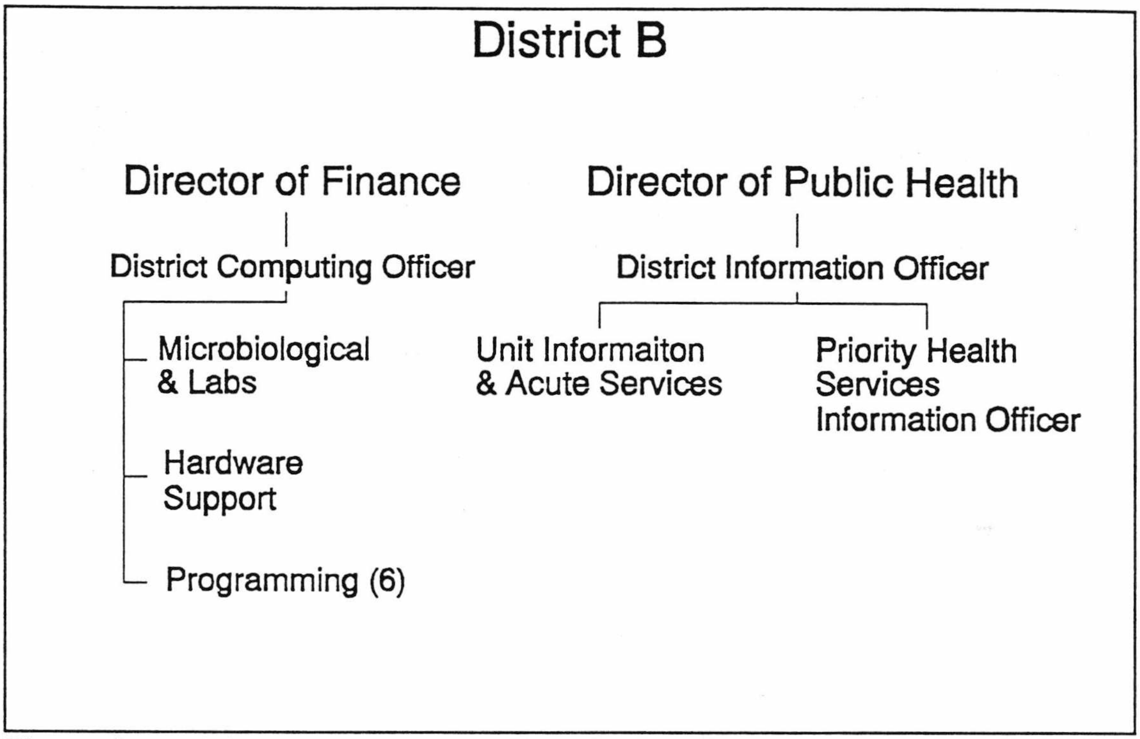

District B
Director of Finance
Director of Public Health
District Computing Officer
District Information Officer
Microbiological & Labs
Unit Informaiton & Acute Services
Priority Health Services Information Officer
Hardware Support
Programming (6)

Figure 9 District B Information Organisation Chart

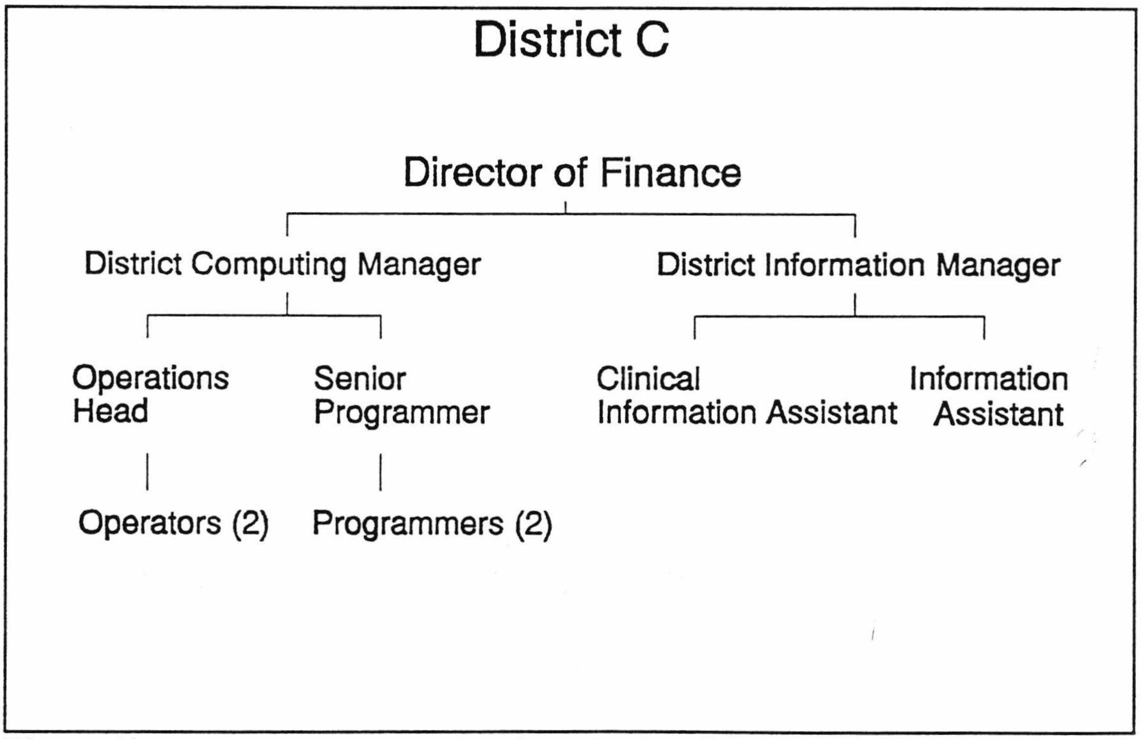


Figure 10 District C Information Organisation Chart

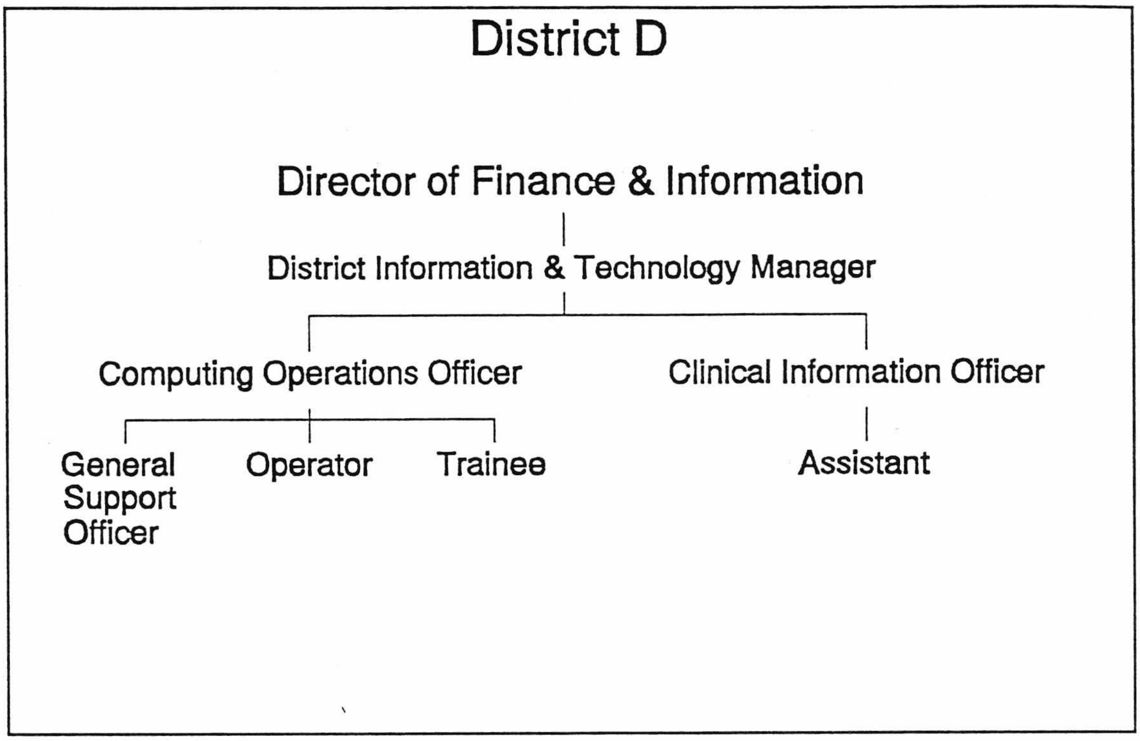

District D
Director of Finance & Information
District Information & Technology Manager
Computing Operations Officer
Clinical Information Officer
General Support Officer
Operator
Trainee
Assistant

Figure 11 District D Information Organisation Chart

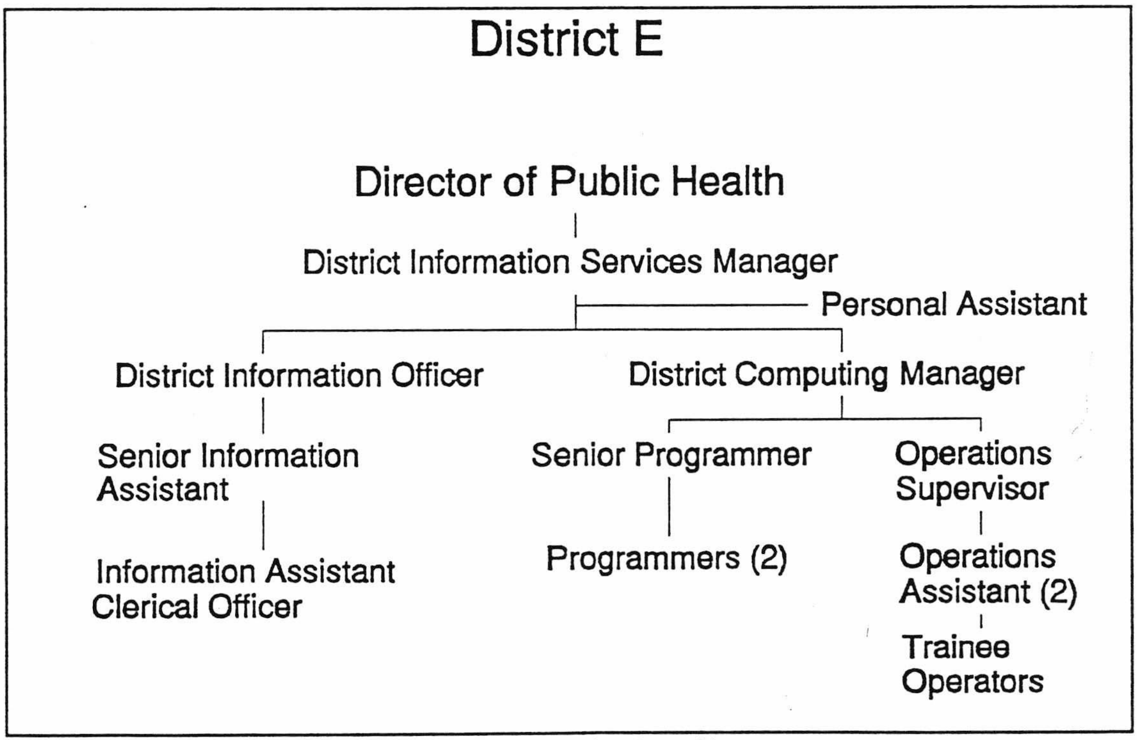


Figure 12 District E Information Organisation Chart

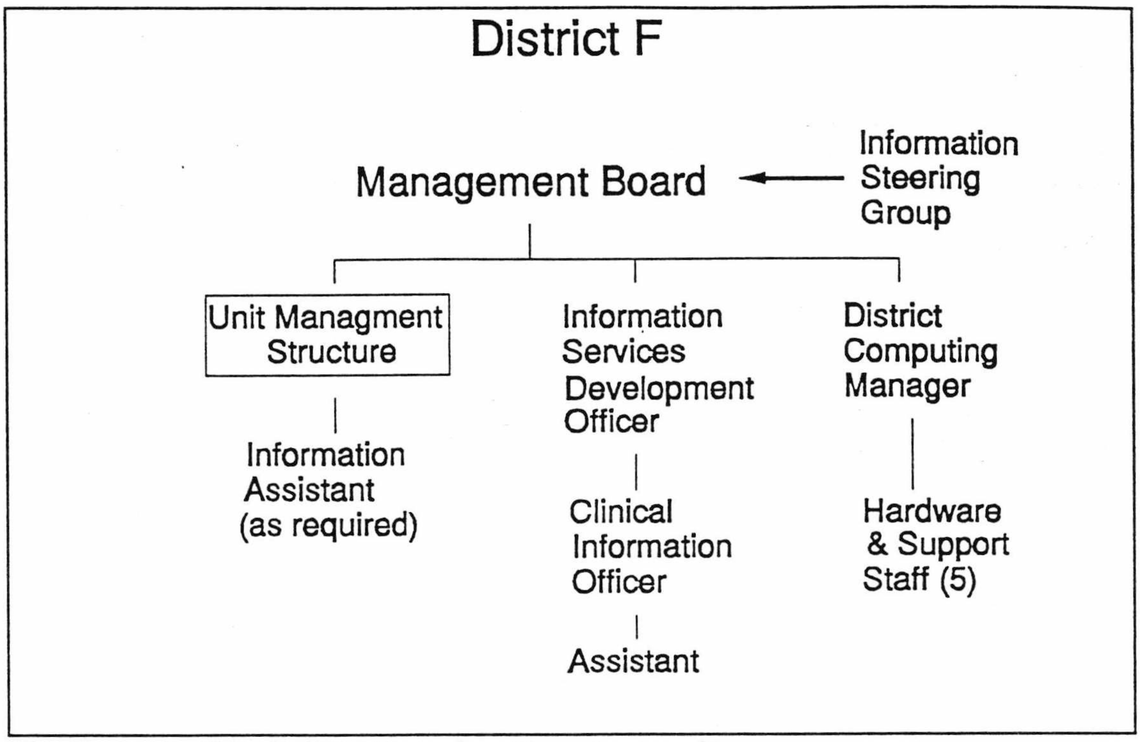

District F
Management Board
Information Steering Group
Unit Managment Structure
Information Services Development Officer
District Computing Manager
Information Assistant (as required)
Clinical Information Officer
Hardware & Support Staff (5)
Assistant

Figure 13 District E Information Organisation Chart

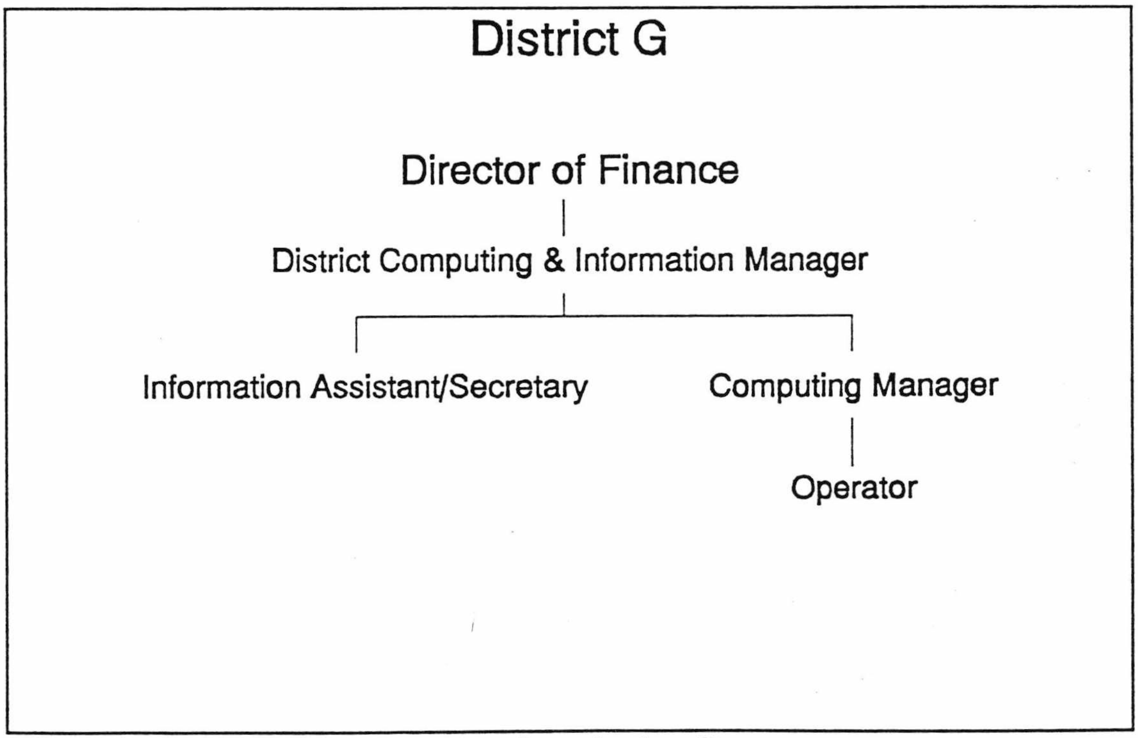

District G
Director of Finance
District Computing & Information Manager
Information Assistant/Secretary
Computing Manager
Operator

Figure 14 District G Information Organisation Chart

APPENDIX THREE

Real Life Information Request

This was a 'real life' request made of the Information Manager in District D.

Provide a report indicating the age and geographic spread of all patients who died of cardia-vascular failure analyised by consultant.

This was then converted into the following:

```
LIST          K-NAME-AND-ADDRESS.SURNAME
              K-NAME-AND-ADDRESS.POSTCODE
SORTED-BY K-PATIENT-DIAG.CONSULTANT-CODE

FOR           K-DIAGNOSIS.STD-DIAGNOSIS-CODE = 48V

THRU          K-DIAGS-IDENT
              K-DIAG-GIVEN
              K-CON-CARE-SPELL
              K-PAT-DET-SPELL
```

APPENDIX FOUR

Short Questionnaire Used In Field Study

Short Questionnaire:

I would be grateful if you would be kind enough to complete this questionnaire. I am a research student at The University Of Kent in Canterbury trying to analyse the workings and results of the Korner Reports and Information flows within this District. This survey will form an important part of that research and I would be grateful if you would find time to complete this today.

Any information that you give will be kept in the STRICTEST CONFIDENCE and will only be seen by me. I would be grateful if you would NOT put your name on the form to maintain this confidentiality.

Please could you ensure that you complete all of the questions and answer them as fully as possible. I invite you to feel free to scribble any comments or answers on the form to express yourself more fully. At the end could you check that all of the questions have been answered and then fold the paper, staple if possible and return it to me.

If you have any queries or questions please free to ask.

Many thanks in advance.

Richard.

Instructions:

Some questions have a scale of possible answers, ranging from one extreme on the left to another extreme on the right.

Example:

In your opinion, what is the weather like today?

Poor	1	2	3	4	5	6	7	[✓ 8]	9	10	Good

Please circle a number to show your answer, for example circle 1 if the weather is poor, 10 if it is good and 5 if it is fair etc.

The other questions require you to write in your comments or answer. If for any reason you don't know, just write don't know.

Example:

In your opinion does the weather change in relation to whether there is an 'r' in the month? Dont know!

Please Turn Over

How long have you been working with computers?

Years:

How was this experience gained? For each job please indicate the job title and description of the experience gained: (if none write NONE)

Job Title Description

How long have you worked within the NHS?

Years:

How long have you worked in NHS information provision/storage? (whether computer based or not)

Years:

How long have you worked with PAS (Patient Administration System) and its given interface?

0: No Contact

Years:

Please Turn Over

What training did you receive for the use of PAS?

Out of this training, approximately what proportion of time was devoted to:

PAS Operation/Interrogation eg. Query Master & Report Master: %

Contextual training, eg. the role of PAS, information use in the HS: %

General computer knowledge and skills: %

100 %

=====

How effective was this training in each of these areas?

PAS Operation:

Effective	1	2	3	4	5	6	7	8	9	10	Ineffective

Contextual Training:

Effective	1	2	3	4	5	6	7	8	9	10	Ineffective

General Computer Knowledge

Effective	1	2	3	4	5	6	7	8	9	10	Ineffective

Please Turn Over

How would you rate your confidence in PAS in terms of the following:
(Please indicate a comment where necessary)

RELIABILITY

Little Confidence	1	2	3	4	5	6	7	8	9	10	Complete Confidence

Comment:

USABILITY

Little Confidence	1	2	3	4	5	6	7	8	9	10	Complete Confidence

Comment:

COMPLETENESS

Little Confidence	1	2	3	4	5	6	7	8	9	10	Complete Confidence

Comment:

DATA ACCURACY

Little Confidence	1	2	3	4	5	6	7	8	9	10	Complete Confidence

Comment:

Please Turn Over

To what extent does your work depend on PAS?

Essential 1 2 3 4 5 6 7 8 9 10

How would you rate the level of pressure of your job?

Low 1 2 3 4 5 6 7 8 9 10 High

How do you feel that you are coping with the current workload?

Insufficiently 1 2 3 4 5 6 7 8 9 10 Sufficiently

Finally if there is anything else that you would like to express or add to the above answers, please feel free to write below.

Many thanks,

APPENDIX FIVE

An Example IPREC Form

Admission

Religion

AC	-	Armenian Catholic	BAP	-	Baptist
BUD	-	Buddhist	C/E	-	Church of England
C/G	-	Church of God	CH	-	Christadelphian
CHSC	-	Christian Scientist	C/I	-	Church of Ireland
C/S	-	Church of Scotland	C/W	-	Church of Wales
HIN	-	Hindu	JEW	-	Jewish
JWIT	-	Jehova's Witness	METH	-	Methodist
MORM	-	Mormon	MUS	-	Muslim
N/C	-	Noncomformist	NONE	-	None
OF	-	Other Free Church	ORTH	-	Orthodox
OTH	-	Other	PB	-	Plymouth Bretheren
PEN	-	Pentecostal	PRES	-	Presbyterian
QUAK	-	Quaker	R/C	-	Roman Catholic
RAST	-	Rastafarian	SALV	-	Salvation Army
SIKH	-	Sikh	URC	-	United Reformed
UN	-	Unitarian			

Admission Method

11 - Elective - Waiting List
12 - Elective - Booked
13 - Elective - Planned
21 - Emergency - A & E Dept. in district
22 - Emergency - GP
23 - Emergency - Bed bureau
24 - Emergency - Consultant outpatient clinic
25 - Emergency - Consultant domiciliary visit
28 - Emergency - Other means
31 - Maternity - Antepartum
32 - Maternity - Postnatal
81 - Other - Patients from institution in another district
82 - Other - Baby born in a hospital in the district
83 - Baby born on the way to the hospital

Code 31 or 32 may only be used for females. Code 81 may only be used if the admission source is 51, 52 or 53. Code 82 may only be used if the admission date is the date of birth. Code 83 may only be used if the admission date is the same as or one day after the date of birth.

Admission Source

19 - Usual residence
29 - Temporary residence
39 - Penal establishment
49 - Special hospital
51 - NHS hospital - General ward/ A & E
52 - NHS hospital - Maternity ward
53 - NHS hospital - Mentally ill / Handicapped ward
69 - Local authority accomodation / Foster care
79 - No previous address - Born in / on way to hospital
89 - Other

Codes 51 to 53 are only used with admission method 81.
Code 79 is only used with admission methods 82 and 83.

Admin Category

1 - NHS, 2 - Private, 3 - Amenity
Car. of Detention
0 - Not formally detained
1 - Formally detained under 1983 Health Act Pt.2
2 - Formally detained under 1983 Health Act Pt. 3 or other act.

Intended Management

1 - Hospital stay of at least one night.
2 - No overnight stay
3 - Planned sequence of admissions of at least one night
4 - Regular admission for a sequence of days when patient returns home for remainder of 24 hours.
5 - Regular admission for a sequence of nights when patient returns home for remainder of 24 hours.
6 - Baby born in / on the way to hospital.

Discharge

Discharge Method

1 - Discharged on medical advice
2 - Discharged by self or relative
3 - Discharged by mental health review tribunal.
4 - Dead.
5 - Stilborn.

Destination

19 - Usual residence.
29 - Temporary residence.
39 - Penal establishment, court or police station.
49 - Special hospital.
51 - NHS hospital outside district - General ward
52 - NHS hospital outside district - Maternity or neonates ward.
53 - NHS hospital outside district - Mentally ill / Mentally handicapped ward
69 - Local authority residential accomodation.
79 - Dead.
89 - Other icluding non NHS hospital, nursing home.

Specialty Codes

Surgical specialties

100	General surgery	500	Obstetrics & Gynacology
101	Urology	* 501	Obstetrics for patients usinghospital bed or delivery facilities.
110	Trauma & Orthopaedics	* 502	Gynaecology
120	ENT		
130	Opthalmology		
140	Oral surgery	600	General Practice
141	Restorative Dentistry		
142	Paediatric Dentistry	* 610	Maternity function
143	Orthodontics	* 620	Other than maternity
150	Neurosurgery		
160	Plastic Surgery		
170	Cardiothoracic Surgery		
171	Paediatric Surgery		
180	Accident & Emergency		
190	Anaesthetics		

Medical specialties

300	General Medicine
301	Gastroenterology
302	Endrocrinology
303	Haematology (Clinical)
304	Clinical Physiology
305	Clinical Pharmacology
310	Audiological Medicine
311	Clinical Genetics
320	Cardiology
330	Dermatology
340	Thoracic Medicine
350	Infectious Diseases
360	Genito-Urinary Medicine
361	Nephrology
370	Medical Oncology
371	Nuclear Medicine
400	Neurology
401	Clinical Neuro-Physiology
410	Rheumatology
420	Paediatrics
421	Paediatric Neurology
430	Geriatric Medicine

*Derived codes

If record type 21 - 23
should record
500 as 501
600 as 610

If episode order = 1 &
Admit method 31 Or 32
should record
500 as 501
600 as 610

If episode order 1 &
First ward = Maternity
should record
500 as 501
600 as 610

All other instances
must record
500 as 502
500 as 520

Psychiatry

700	Mental Handicap
710	Mental Illness
711	Child & Adolescent Psychiatry
712	Forensic Psychiatry
713	Psychotherapy
800	Radiotherapy
810	Radiology

Pathology

820	General Pathology
821	Blood Translusion
822	Chemical Pathology
823	Haematology
824	Histopathology
830	Immunopathology
831	Medical Microbiololgy
832	Neuropathology
900	Community Medicine
901	Occupational Medicine
SPACES	Other maternity event

Legal Status Transfer

Legal Status

01 Informal - (Cat. of Det. = 0)

Formally detailed under Mental Health Act Part II, (Cat of Det. = 1)

02 = Section 2	03 = Section 3
04 = Section 4	05 = Section 5 (2)
06 = Section 5 (4)	

Formally detained under Mental Health Act Part III,
(Cat. of Det. = 2 for all remaining sections)

07 = Section 35	08 = Section 36
09 = Section 37X/41	10 = Section 37X
11 = Section 37 (4)	12 = Section 38
13 = Section 44	14 = Section 46
15 = Section 47/49	16 = Section 47
17 = Section 48/49	18 = Section 48

Note: 37X excludes section 37 (4)

Formally detained under Part X:

19 = Section 135 20 = Section 136

21 = Other section

Formally detained under:

30	Previous legislation
31	Criminal procedure (Insanity) Act 1964
32	Other acts

Mental Category 1

1 = Mental Illness	2 = Psychopathic disorder
3 = Mental impairment	4 = Severe mental impairment
8 = Others	& = Not known
- = Not applicable	

K.[illegible].R.1

PATIENT AND ATD DETAILS

Patient Number ________ N.H.S. Number ________

Surname ________ Forenames ________

Sex □ Date of Birth / /

M = Male
F = Female
I = Indeterminate

Address ________

Postcode ________ Post Town ________ Marital Status □ Religion ________

G.P. ________

SPELL NUMBER ________ ADMISSION DATE / /

ADMISSION DETAILS

HOSPITAL ________

WARD ________

CONSULTANT ________

SPECIALTY ________

Admission Source	Admission Method	Admin. Category	Cat. of Detention	Intended Mgmt.	Waiting List Date	Age on Admission
□	□	□	□	□	/ /	□

DISCHARGE DETAILS

Date of Discharge	Discharge Method	Destination
/ /	□	□

TRANSFER TO DETAILS

Start Date	Start Time	Consultant Code	Specialty Code	Ward	Hospital
/ /					
/ /					
/ /					
/ /					
/ /					
/ /					
/ /					
/ /					
/ /					
/ /					

PATIENT CATEGORY TRANSFER

Start Date	Admin. Category
/ /	□

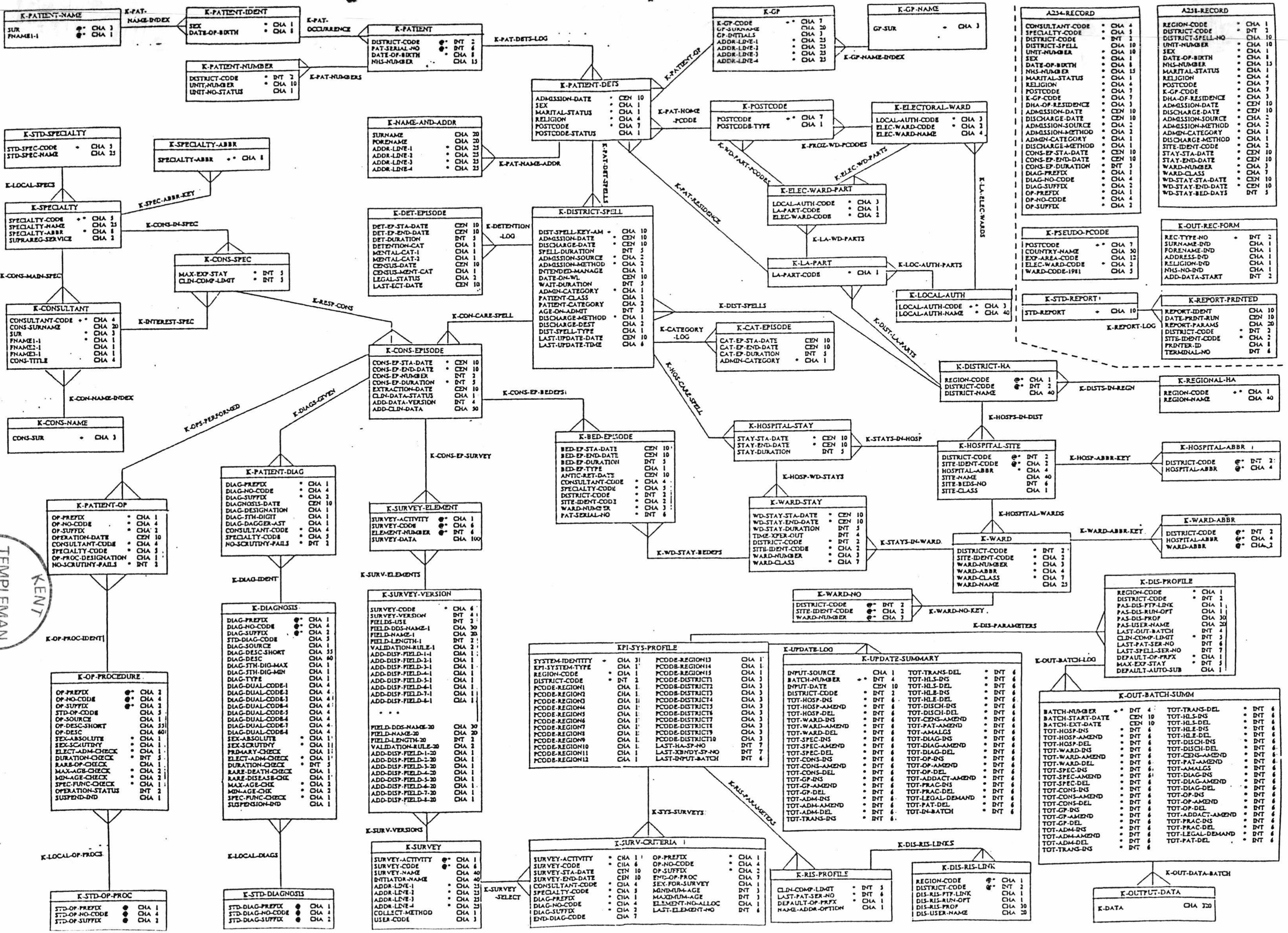
LEGAL STATUS TRANSFER

Start Date	End Date	Legal Status	Mental 1st.	Category 2nd.
/ /	/ /			
/ /	/ /			
/ /	/ /			
/ /	/ /			
/ /	/ /			

APPENDIX SIX

DIS Datamodel Query View Diagram

QUERY VIEW KOVWT2



K-PATIENT-NAME
SUR CHA 3
FNAME1-1 CHA 1
K-PAT-NAME-INDEX
K-PATIENT-IDENT
SEX CHA 1
DATE-OF-BIRTH CHA 8
K-PAT-OCCURRENCE
K-PATIENT
DISTRICT-CODE INT 2
PAT-SERIAL-NO INT 6
DATE-OF-BIRTH CHA 8
NHS-NUMBER CHA 15
K-PAT-DETS-LOG
K-PATIENT-NUMBER
DISTRICT-CODE INT 2
UNIT-NUMBER CHA 10
UNIT-NO-STATUS CHA 1
K-PAT-NUMBERS
K-GP
K-GP-CODE CHA 7
GP-SURNAME CHA 20
GP-INITIALS CHA 3
ADDR-LINE-1 CHA 25
ADDR-LINE-2 CHA 25
ADDR-LINE-3 CHA 25
ADDR-LINE-4 CHA 25
K-GP-NAME
GP-SUR CHA 3
K-GP-NAME-INDEX
K-PATIENT-GP
A234-RECORD
CONSULTANT-CODE CHA 4
SPECIALTY-CODE CHA 3
DISTRICT-CODE INT 2
DISTRICT-SPELL CHA 10
UNIT-NUMBER CHA 10
SEX CHA 1
DATE-OF-BIRTH CHA 8
NHS-NUMBER CHA 15
MARITAL-STATUS CHA 1
RELIGION CHA 4
POSTCODE CHA 7
K-GP-CODE CHA 7
DHA-OF-RESIDENCE CHA 3
ADMISSION-DATE CEN 10
DISCHARGE-DATE CEN 10
ADMISSION-SOURCE CHA 2
ADMISSION-METHOD CHA 2
ADMIN-CATEGORY CHA 1
DISCHARGE-METHOD CHA 1
CONS-EP-STA-DATE CEN 10
CONS-EP-END-DATE CEN 10
CONS-EP-DURATION INT 5
DIAG-PREFIX CHA 1
DIAG-NO-CODE CHA 4
DIAG-SUFFIX CHA 2
OP-PREFIX CHA 1
OP-NO-CODE CHA 4
OP-SUFFIX CHA 2
A258-RECORD
REGION-CODE CHA 1
DISTRICT-CODE INT 2
DISTRICT-SPELL-NO CHA 10
UNIT-NUMBER CHA 10
SEX CHA 1
DATE-OF-BIRTH CHA 8
NHS-NUMBER CHA 15
MARITAL-STATUS CHA 1
RELIGION CHA 4
POSTCODE CHA 7
K-GP-CODE CHA 7
DHA-OF-RESIDENCE CHA 3
ADMISSION-DATE CEN 10
DISCHARGE-DATE CEN 10
ADMISSION-SOURCE CHA 2
ADMISSION-METHOD CHA 2
ADMIN-CATEGORY CHA 1
DISCHARGE-METHOD CHA 1
SITE-IDENT-CODE CHA 2
STAY-STA-DATE CEN 10
STAY-END-DATE CEN 10
WARD-NUMBER CHA 3
WARD-CLASS CHA 7
WD-STAY-STA-DATE CEN 10
WD-STAY-END-DATE CEN 10
WD-STAY-BED-DAYS INT 5
K-PATIENT-DETS
ADMISSION-DATE CEN 10
SEX CHA 1
MARITAL-STATUS CHA 1
RELIGION CHA 4
POSTCODE CHA 7
POSTCODE-STATUS CHA 1
K-PAT-HOME-PCODE
K-POSTCODE
POSTCODE CHA 7
POSTCODE-TYPE CHA 1
K-ELECTORAL-WARD
LOCAL-AUTH-CODE CHA 3
ELEC-WARD-CODE CHA 2
ELEC-WARD-NAME CHA 4
K-PROZ-WD-PCODES
K-WD-PART-PCODES
K-ELEC-WD-PARTS
K-ELEC-WARD-PART
LOCAL-AUTH-CODE CHA 3
LA-PART-CODE CHA 1
ELEC-WARD-CODE CHA 2
K-LA-WD-PARTS
K-LA-PART
LA-PART-CODE CHA 1
K-LOC-AUTH-PARTS
K-LA-ELEC-WARDS
K-LOCAL-AUTH
LOCAL-AUTH-CODE CHA 3
LOCAL-AUTH-NAME CHA 40
K-PSEUDO-PCODE
POSTCODE CHA 7
COUNTRY-NAME CHA 30
EXP-AREA-CODE CHA 12
ELEC-WARD-CODE CHA 2
WARD-CODE-1981 CHA 5
K-OUT-REC-FORM
REC-TYPE-NO INT 2
SURNAME-IND CHA 1
FORENAME-IND CHA 1
ADDRESS-IND CHA 1
RELIGION-IND CHA 1
NHS-NO-IND CHA 1
ADD-DATA-START INT 2
K-STD-REPORT
STD-REPORT CHA 10
K-REPORT-LOG
K-REPORT-PRINTED
REPORT-IDENT CHA 10
DATE-PRINT-RUN CEN 10
REPORT-PARAMS CHA 20
DISTRICT-CODE INT 2
SITE-IDENT-CODE CHA 2
PRINTER-ID CHA 8
TERMINAL-NO INT 6
K-NAME-AND-ADDR
SURNAME CHA 20
FORENAME CHA 20
ADDR-LINE-1 CHA 25
ADDR-LINE-2 CHA 25
ADDR-LINE-3 CHA 25
ADDR-LINE-4 CHA 25
K-PAT-NAME-ADDR
K-PAT-DET-SPELLS
K-PAT-RESIDENCE
K-STD-SPECIALTY
STD-SPEC-CODE CHA 3
STD-SPEC-NAME CHA 25
K-SPECIALTY-ABBR
SPECIALTY-ABBR CHA 8
K-LOCAL-SPECS
K-SPEC-ABBR-KEY
K-SPECIALTY
SPECIALTY-CODE CHA 3
SPECIALTY-NAME CHA 25
SPECIALTY-ABBR CHA 8
SUPRAREG-SERVICE CHA 2
K-CONS-IN-SPEC
K-CONS-SPEC
MAX-EXP-STAY INT 5
CLIN-COMP-LIMIT INT 3
K-CONS-MAIN-SPEC
K-CONSULTANT
CONSULTANT-CODE CHA 4
CONS-SURNAME CHA 20
SUR CHA 3
FNAME1-1 CHA 1
FNAME2-1 CHA 1
FNAME3-1 CHA 1
CONS-TITLE CHA 4
K-INTEREST-SPEC
K-CON-NAME-INDEX
K-CONS-NAME
CONS-SUR CHA 3
K-DET-EPISODE
DET-EP-STA-DATE CEN 10
DET-EP-END-DATE CEN 10
DET-DURATION INT 5
DETENTION-CAT CHA 1
MENTAL-CAT-1 CHA 1
MENTAL-CAT-2 CHA 1
CENSUS-DATE CEN 10
CENSUS-MENT-CAT CHA 1
LEGAL-STATUS CHA 2
LAST-ECT-DATE CEN 10
K-DETENTION-LOG
K-DISTRICT-SPELL
DIST-SPELL-KEY-AM CHA 10
ADMISSION-DATE CEN 10
DISCHARGE-DATE CEN 10
SPELL-DURATION INT 5
ADMISSION-SOURCE CHA 2
ADMISSION-METHOD CHA 2
INTENDED-MANAGE CHA 1
DATE-ON-WL CEN 10
WAIT-DURATION INT 5
ADMIN-CATEGORY CHA 1
PATIENT-CLASS CHA 1
PATIENT-CATEGORY CHA 2
AGE-ON-ADMIT INT 3
DISCHARGE-METHOD CHA 1
DISCHARGE-DEST CHA 2
DIST-SPELL-TYPE CHA 1
LAST-UPDATE-DATE CEN 10
LAST-UPDATE-TIME CHA 6
K-DIST-SPELLS
K-CATEGORY-LOG
K-CAT-EPISODE
CAT-EP-STA-DATE CEN 10
CAT-EP-END-DATE CEN 10
CAT-EP-DURATION INT 5
ADMIN-CATEGORY CHA 1
K-DIST-LA-PARTS
K-DISTRICT-HA
REGION-CODE CHA 1
DISTRICT-CODE INT 2
DISTRICT-NAME CHA 40
K-DISTS-IN-REGN
K-REGIONAL-HA
REGION-CODE CHA 1
REGION-NAME CHA 40
K-RESP-CONS
K-CON-CARE-SPELL
K-CONS-EPISODE
CONS-EP-STA-DATE CEN 10
CONS-EP-END-DATE CEN 10
CONS-EP-NUMBER INT 2
CONS-EP-DURATION INT 5
EXTRACTION-DATE CEN 10
CLIN-DATA-STATUS CHA 1
ADD-DATA-VERSION INT 4
ADD-CLIN-DATA CHA 50
K-CONS-EP-BEDEPS
K-HOS-CARE-SPELL
K-HOSPS-IN-DIST
K-OPS-PERFORMED
K-DIAGS-GIVEN
K-BED-EPISODE
BED-EP-STA-DATE CEN 10
BED-EP-END-DATE CEN 10
BED-EP-DURATION INT 5
BED-EP-TYPE CHA 1
ANTIC-RET-DATE CEN 10
CONSULTANT-CODE CHA 4
SPECIALTY-CODE CHA 3
DISTRICT-CODE INT 2
SITE-IDENT-CODE CHA 2
WARD-NUMBER CHA 3
PAT-SERIAL-NO INT 6
K-HOSPITAL-STAY
STAY-STA-DATE CEN 10
STAY-END-DATE CEN 10
STAY-DURATION INT 5
K-STAYS-IN-HOSP
K-HOSPITAL-SITE
DISTRICT-CODE INT 2
SITE-IDENT-CODE CHA 2
HOSPITAL-ABBR CHA 4
SITE-NAME CHA 40
SITE-BEDS-NO INT 6
SITE-CLASS CHA 1
K-HOSP-ABBR-KEY
K-HOSPITAL-ABBR
DISTRICT-CODE INT 2
HOSPITAL-ABBR CHA 4
K-HOSP-WD-STAYS
K-CONS-EP-SURVEY
K-PATIENT-DIAG
DIAG-PREFIX CHA 1
DIAG-NO-CODE CHA 4
DIAG-SUFFIX CHA 2
DIAGNOSIS-DATE CEN 10
DIAG-DESIGNATION CHA 1
DIAG-5TH-DIGIT CHA 1
DIAG-DAGGER-AST CHA 1
CONSULTANT-CODE CHA 4
SPECIALTY-CODE CHA 3
NO-SCRUTINY-FAILS INT 2
K-PATIENT-OP
OP-PREFIX CHA 1
OP-NO-CODE CHA 4
OP-SUFFIX CHA 2
OPERATION-DATE CEN 10
CONSULTANT-CODE CHA 4
SPECIALTY-CODE CHA 3
OP-PROC-DESIGNATION CHA 1
NO-SCRUTINY-FAILS INT 2
K-SURVEY-ELEMENT
SURVEY-ACTIVITY CHA 1
SURVEY-CODE CHA 6
ELEMENT-NUMBER INT 4
SURVEY-DATA CHA 100
K-WARD-STAY
WD-STAY-STA-DATE CEN 10
WD-STAY-END-DATE CEN 10
WD-STAY-DURATION INT 5
TIME-XFER-OUT INT 4
DISTRICT-CODE INT 2
SITE-IDENT-CODE CHA 2
WARD-NUMBER CHA 3
WARD-CLASS CHA 7
K-HOSPITAL-WARDS
K-STAYS-IN-WARD
K-WARD
DISTRICT-CODE INT 2
SITE-IDENT-CODE CHA 2
WARD-NUMBER CHA 3
WARD-ABBR CHA 4
WARD-CLASS CHA 7
WARD-NAME CHA 25
K-WARD-ABBR-KEY
K-WARD-ABBR
DISTRICT-CODE INT 2
HOSPITAL-ABBR CHA 4
WARD-ABBR CHA 2
K-WD-STAY-BEDEPS
K-DIAG-IDENT
K-SURV-ELEMENTS
K-WARD-NO
DISTRICT-CODE INT 2
SITE-IDENT-CODE CHA 2
WARD-NUMBER CHA 3
K-WARD-NO-KEY
K-DIS-PROFILE
REGION-CODE CHA 1
DISTRICT-CODE INT 2
PAS-DIS-FTP-LINK CHA 1
PAS-DIS-RUN-OPT CHA 1
PAS-DIS-PROF CHA 30
PAS-USER-NAME CHA 20
LAST-OUT-BATCH INT 4
CLIN-COMP-LIMIT INT 3
LAST-PAT-SER-NO INT 6
LAST-SPELL-SER-NO INT 7
DEFAULT-OP-PRFX CHA 1
MAX-EXP-STAY INT 5
DEFAULT-AUTO-SUB CHA 1
K-DIS-PARAMETERS
K-OP-PROC-IDENT
K-DIAGNOSIS
DIAG-PREFIX CHA 1
DIAG-NO-CODE CHA 4
DIAG-SUFFIX CHA 2
STD-DIAG-CODE CHA 5
DIAG-SOURCE CHA 1
DIAG-DESC-SHORT CHA 35
DIAG-DESC CHA 60
DIAG-5TH-DIG-MAX CHA 1
DIAG-5TH-DIG-MIN CHA 1
DIAG-TYPE CHA 1
DIAG-DUAL-CODE-1 CHA 4
DIAG-DUAL-CODE-2 CHA 4
DIAG-DUAL-CODE-3 CHA 4
DIAG-DUAL-CODE-4 CHA 4
DIAG-DUAL-CODE-5 CHA 4
DIAG-DUAL-CODE-6 CHA 4
DIAG-DUAL-CODE-7 CHA 4
DIAG-DUAL-CODE-8 CHA 4
SEX-ABSOLUTE CHA 1
SEX-SCRUTINY CHA 1
PRIMARY-CHECK CHA 1
ELECT-ADM-CHECK CHA 1
DURATION-CHECK CHA 1
RARE-DEATH-CHECK CHA 1
RARE-DISEASE-CHK CHA 1
MAX-AGE-CHK CHA 3
MIN-AGE-CHK CHA 3
SPEC-FUNC-CHECK CHA 1
SUSPENSION-IND CHA 1
K-SURVEY-VERSION
SURVEY-CODE CHA 6
SURVEY-VERSION INT 4
FIELDS-USE INT 2
FIELD-DDS-NAME-1 CHA 30
FIELD-NAME-1 CHA 20
FIELD-LENGTH-1 INT 2
VALIDATION-RULE-1 CHA 2
ADD-DISP-FIELD-1-1 CHA 1
ADD-DISP-FIELD-2-1 CHA 1
ADD-DISP-FIELD-3-1 CHA 1
ADD-DISP-FIELD-4-1 CHA 1
ADD-DISP-FIELD-5-1 CHA 1
ADD-DISP-FIELD-6-1 CHA 1
ADD-DISP-FIELD-7-1 CHA 1
ADD-DISP-FIELD-8-1 CHA 1
...
FIELD-DDS-NAME-20 CHA 30
FIELD-NAME-20 CHA 20
FIELD-LENGTH-20 INT 2
VALIDATION-RULE-20 CHA 2
ADD-DISP-FIELD-1-20 CHA 1
ADD-DISP-FIELD-2-20 CHA 1
ADD-DISP-FIELD-3-20 CHA 1
ADD-DISP-FIELD-4-20 CHA 1
ADD-DISP-FIELD-5-20 CHA 1
ADD-DISP-FIELD-6-20 CHA 1
ADD-DISP-FIELD-7-20 CHA 1
ADD-DISP-FIELD-8-20 CHA 1
K-OP-PROCEDURE
OP-PREFIX CHA 1
OP-NO-CODE CHA 4
OP-SUFFIX CHA 2
STD-OP-CODE CHA 5
OP-SOURCE CHA 1
OP-DESC-SHORT CHA 35
OP-DESC CHA 60
SEX-ABSOLUTE CHA 1
SEX-SCRUTINY CHA 1
ELECT-ADM-CHECK CHA 1
DURATION-CHECK INT 5
RARE-OP-CHECK CHA 1
MAX-AGE-CHECK CHA 3
MIN-AGE-CHECK CHA 3
SPEC-FUNC-CHECK CHA 1
OPERATION-STATUS INT 2
SUSPEND-IND CHA 1
KPI-SYS-PROFILE
SYSTEM-IDENTITY CHA 31
KPI-SYS-TYPE CHA 1
REGION-CODE CHA 1
DISTRICT-CODE INT 2
PCODE-REGION1 CHA 1
PCODE-REGION2 CHA 1
PCODE-REGION3 CHA 1
PCODE-REGION4 CHA 1
PCODE-REGION5 CHA 1
PCODE-REGION6 CHA 1
PCODE-REGION7 CHA 1
PCODE-REGION8 CHA 1
PCODE-REGION9 CHA 1
PCODE-REGION10 CHA 1
PCODE-REGION11 CHA 1
PCODE-REGION12 CHA 1
PCODE-REGION13 CHA 1
PCODE-REGION14 CHA 1
PCODE-REGION15 CHA 1
PCODE-DISTRICT1 CHA 3
PCODE-DISTRICT2 CHA 3
PCODE-DISTRICT3 CHA 3
PCODE-DISTRICT4 CHA 3
PCODE-DISTRICT5 CHA 3
PCODE-DISTRICT6 CHA 3
PCODE-DISTRICT7 CHA 3
PCODE-DISTRICT8 CHA 3
PCODE-DISTRICT9 CHA 3
PCODE-DISTRICT10 CHA 3
LAST-HA-SP-NO INT 7
LAST-XBNDY-SP-NO INT 7
LAST-INPUT-BATCH INT 4
K-UPDATE-LOG
K-UPDATE-SUMMARY
INPUT-SOURCE CHA 1
BATCH-NUMBER INT 4
INPUT-DATE CEN 10
DISTRICT-CODE INT 2
TOT-HOSP-INS INT 6
TOT-HOSP-AMEND INT 6
TOT-HOSP-DEL INT 6
TOT-WARD-INS INT 6
TOT-WARD-AMEND INT 6
TOT-WARD-DEL INT 6
TOT-SPEC-INS INT 6
TOT-SPEC-AMEND INT 6
TOT-SPEC-DEL INT 6
TOT-CONS-INS INT 6
TOT-CONS-AMEND INT 6
TOT-CONS-DEL INT 6
TOT-GP-INS INT 6
TOT-GP-AMEND INT 6
TOT-GP-DEL INT 6
TOT-ADM-INS INT 6
TOT-ADM-AMEND INT 6
TOT-ADM-DEL INT 6
TOT-TRANS-INS INT 6
TOT-TRANS-DEL INT 6
TOT-HLS-INS INT 6
TOT-HLS-DEL INT 6
TOT-HLE-INS INT 6
TOT-HLE-DEL INT 6
TOT-DISCH-INS INT 6
TOT-DISCH-DEL INT 6
TOT-CENS-AMEND INT 6
TOT-PAT-AMEND INT 6
TOT-AMALGS INT 6
TOT-DIAG-INS INT 6
TOT-DIAG-AMEND INT 6
TOT-DIAG-DEL INT 6
TOT-OP-INS INT 6
TOT-OP-AMEND INT 6
TOT-OP-DEL INT 6
TOT-ADDACT-AMEND INT 6
TOT-PRAC-INS INT 6
TOT-PRAC-DEL INT 6
TOT-LEGAL-DEMAND INT 6
TOT-PAT-DEL INT 6
TOT-IN-BATCH INT 6
K-OUT-BATCH-LOG
K-OUT-BATCH-SUMM
BATCH-NUMBER INT 4
BATCH-START-DATE CEN 10
BATCH-EXT-DATE CEN 10
TOT-HOSP-INS INT 6
TOT-HOSP-AMEND INT 6
TOT-HOSP-DEL INT 6
TOT-WARD-INS INT 6
TOT-WARD-AMEND INT 6
TOT-WARD-DEL INT 6
TOT-SPEC-INS INT 6
TOT-SPEC-AMEND INT 6
TOT-SPEC-DEL INT 6
TOT-CONS-INS INT 6
TOT-CONS-AMEND INT 6
TOT-CONS-DEL INT 6
TOT-GP-INS INT 6
TOT-GP-AMEND INT 6
TOT-GP-DEL INT 6
TOT-ADM-INS INT 6
TOT-ADM-AMEND INT 6
TOT-ADM-DEL INT 6
TOT-TRANS-INS INT 6
TOT-TRANS-DEL INT 6
TOT-HLS-INS INT 6
TOT-HLS-DEL INT 6
TOT-HLE-INS INT 6
TOT-HLE-DEL INT 6
TOT-DISCH-INS INT 6
TOT-DISCH-DEL INT 6
TOT-CENS-AMEND INT 6
TOT-PAT-AMEND INT 6
TOT-AMALGS INT 6
TOT-DIAG-INS INT 6
TOT-DIAG-AMEND INT 6
TOT-DIAG-DEL INT 6
TOT-OP-INS INT 6
TOT-OP-AMEND INT 6
TOT-OP-DEL INT 6
TOT-ADDACT-AMEND INT 6
TOT-PRAC-INS INT 6
TOT-PRAC-DEL INT 6
TOT-LEGAL-DEMAND INT 6
TOT-PAT-DEL INT 6
K-SYS-SURVEYS
K-RIS-PARAMETERS
K-DIS-RIS-LINKS
K-SURV-VERSIONS
K-LOCAL-OP-PROCS
K-LOCAL-DIAGS
K-STD-OP-PROC
STD-OP-PREFIX CHA 1
STD-OP-NO-CODE CHA 4
STD-OP-SUFFIX CHA 2
K-STD-DIAGNOSIS
STD-DIAG-PREFIX CHA 1
STD-DIAG-NO-CODE CHA 4
STD-DIAG-SUFFIX CHA 2
K-SURVEY
SURVEY-ACTIVITY CHA 1
SURVEY-CODE CHA 6
SURVEY-NAME CHA 40
INITIATOR-NAME CHA 40
ADDR-LINE-1 CHA 25
ADDR-LINE-2 CHA 25
ADDR-LINE-3 CHA 25
ADDR-LINE-4 CHA 25
COLLECT-METHOD CHA 1
USER-CODE CHA 3
K-SURVEY-SELECT
K-SURV-CRITERIA
SURVEY-ACTIVITY CHA 1
SURVEY-CODE CHA 6
SURVEY-STA-DATE CEN 10
SURVEY-END-DATE CEN 10
CONSULTANT-CODE CHA 4
SPECIALTY-CODE CHA 3
DIAG-PREFIX CHA 1
DIAG-SUFFIX CHA 2
END-DIAG-CODE CHA 7
OP-PREFIX CHA 1
OP-NO-CODE CHA 4
OP-SUFFIX CHA 2
END-OP-PROC CHA 7
SEX-FOR-SURVEY CHA 1
MINIMUM-AGE INT 3
MAXIMUM-AGE INT 3
ELEMENT-NO-ALLOC INT 4
LAST-ELEMENT-NO INT 4
K-RIS-PROFILE
CLIN-COMP-LIMIT INT 3
LAST-PAT-SER-NO INT 6
DEFAULT-OP-PRFX CHA 1
NAME-ADDR-OPTION CHA 1
K-DIS-RIS-LINK
REGION-CODE CHA 1
DISTRICT-CODE INT 2
DIS-RIS-FTP-LINK CHA 1
DIS-RIS-RUN-OPT CHA 1
DIS-RIS-PROF CHA 30
DIS-USER-NAME CHA 20
K-OUT-DATA-BATCH
K-OUTPUT-DATA
K-DATA CHA 320