Chapter 1

INTRODUCTION

1.1 OVERVIEW

This thesis reports on a school-based project designed to enrich the use of computers in school administration beyond merely providing statistical data for decision makers to providing them with actual recommendations. The project involved the development, trialing and evaluation of an expert system to recommend appropriate subject selection for individual students.

Remus and Kottemann (1986, 2) noted that "making a good decision starts with having or gathering the right information upon which to base a decision". Simple computer systems provide a means of filing data while more complex computer systems can use techniques to sort and interrogate data, but decision makers still need the expertise to ask the right questions and thus gather the right information on which to base decisions. Computer decision support systems, such as expert systems, are designed to incorporate this expertise and provide decision makers with a range of options rather than a range of data.

There is an increasing volume of literature in three related yet distinct areas. (1) Authors associated with computer technology and reporting on, or advocating its use by, practising administrators. (2) An expanding interest in the use of computers by schools to assist administrative practices. Such literature, especially in Australia, has primarily focused on clerical tasks. (3) A growing volume of information about artificial intelligence in general, and expert systems in particular.

Thus far the emphasis in the use of computing to assist a school's administration has been on systems to sort and present data. There appears to be a lack of information, let alone strategies, as to how schools can use sophisticated computer technology as a decision support system. This situation is illustrated by the implementation strategies for the introduction of computer systems into schools, which sometimes include training on how to *operate* the computers, but rarely include training on how to *use* the computers (Bucknall 1994a). The research reported in this thesis aims to make a contribution to this area by designing and implementing a specific decision support system for use in a trial school; that is, the development of a computer program which provides users with a range of options to facilitate decision making.

1.2 BACKGROUND TO THE STUDY

In late 1984, the Northern Territory Department of Education commenced a project to place a dedicated mini-computer network in every government school where the enrolment exceeded 250 students. These computers were installed specifically for use in school administration, and were not for teaching students. All the schools are using a common suite of software which caters for word processing, student personal details, student and staff timetables, student assessment and reporting, and school financial records. Additional programs are available to provide asset registers, library cataloguing and borrowing facilities, and electronic mail services.

The evolution of computer assistance for decision making can be traced through three stages. The first involves *Electronic Data Processing* (EDP) in which tasks are relatively self contained and routine for recording data. The second involves *Management Information Systems* (MIS) which are intended to extract data from which decisions can be made. More recently, *expert systems* are being designed to recommend appropriate decisions. The introduction of the Northern Territory School Information System has reflected the first two of these stages. In most Northern Territory schools there are Electronic Data Processing programs to maintain student, financial and other records; there are some Management Information Systems to generate reports that summarise data. However, there are no computer programs providing decision support systems which provide recommendations rather than data and thus competent decision making remains critically reliant on effective procedures and the individual training, knowledge and expertise of the decision makers. Romiszowski (1987, 17) expressed the view that access to effective and efficient expert systems "would be welcomed by many and would no doubt make itself felt in terms of improved decisions ... the moot point is whether the field of education will attract the investment necessary to develop such special aids".

Over the last decade there has been an increase in the number of Australian secondary schools offering term or semester length curriculum units through a vertical timetable. A number of writers (for example, Middleton 1982, Maxwell *et al.* 1987 and

Marshall *et al.* 1988) have reported on these curriculum and timetabling arrangements in some Australian schools. Fowler's (1993a) more recent comparison of several Northern Territory schools is especially pertinent to the research reported in this thesis. Her main criticisms of a vertical timetable confirm the informal and anecdotal statements, which in part prompted this project, of the difficulties faced by students, parents and teachers keeping track of subject pathways and their consequences. Unit selection is a critical component of the timetabling process, which Johnson (1980, 9) described as "probably the most important single event in the school year ... [which] may easily distort or destroy the curriculum philosophy of the school." The process of recommending appropriate units for students is complex and requires a variety of expertise and thus was considered an appropriate domain for attempting to develop a computer program to provide a decision support system.

1.3 RESEARCH DOMAIN

Research during the late 1950s and early 1960s into artificial intelligence concentrated on discovering the fundamental laws of reasoning which a computer system would then be able to apply. The failure of the early research into artificial intelligence to achieve satisfactory reasoning performance indicated that effective reasoning systems needed to include domain specific knowledge. Expert systems, one of the sub-sets of artificial intelligence, are computer models containing both logic and knowledge data and through their ability to explain their reasoning are a sophisticated example of a decision support system.

The title of this thesis, "Using Artificial Intelligence as a Decision Support System in School Administration", suggests a broader study than in fact has been undertaken. The term artificial intelligence has been used in the title as it appears to be more readily recognised in the wider community, though not necessarily any better understood, than the term expert system. The researcher hopes that the main title indicates a potential role for computers in school administration, and that the sub-title "The Development of an Expert System for Student Subject Selection" indicates an example of that role.

The study of expert systems increased rapidly in many countries during the period 1989-1993. In keeping with other computing developments, comments and predictions for the research field made relatively recently are occasionally already out-of-date, especially in relation to expectations concerning the design and implementation of expert systems. Thus some technical aspects of this thesis are probably already out-of-date, but the researcher anticipates that the central findings will be confirmed rather than contradicted by further developments in expert systems.

One difficulty in researching this developing field has been the breadth of literature ranging from articles in the popular press through to complex mathematical treatises. Further, the perceived variations and contradictions between many publications in describing what would appear to be key tenets of expert systems suggests that attempts by some authors to contribute to this domain still have a long way to go. This thesis has concentrated on non-technical literature with the intention that amateurs in computer system design and construction be able to read a non-technical document.

1.4 PURPOSE OF THE STUDY

The primary purpose of this research project has been to ascertain whether computerised decision support systems, such as expert systems, can be developed to assist in the administration of schools. Reports in the literature suggest they should and thus a central research problem was to demonstrate that they could. Resolution of this problem involved several elements: (1) modelling a specific decision making domain in a school, (2) designing and implementing an expert system to assist decision making in this domain, and (3) evaluating the expert system to validate its recommendations and compare its performance with the current system and human experts. It is hoped that the findings reported in this thesis will encourage school leaders to consider the applicability of computerised decision support systems for this and other areas of responsibility or interest.

In addition to the primary aim, the research had several related objectives, three of which are specified here.

(1) One objective of this research was to explore the degree of involvement that school personnel can have in the development of an expert system. Most computer systems in Northern Territory schools have been developed as standard applications. There have been some attempts to involve representatives from schools in these developments but school personnel have generally lacked a sense of participation and ownership in the new technology (Bucknall 1988). In addition to enhancing product acceptance, it is anticipated that increased staff participation may enrich the development process and outcome. On the other hand, the development of an expert system may be too complex or time consuming to meaningfully involve school personnel.

(2) Another objective was to develop a facility to encapsulate the expertise of leaders in the domain and make it available to guide others. Schools, like many other organisations, are frequently faced with the need for decisions based on the expert knowledge of individuals. The quality of the expertise available in a school at a given time will vary. A variety of methods have been adopted to encapsulate the expert knowledge: for example, keep the expert, train more experts, document the expert's knowledge, and prepare contingency plans in the event that the expert is not available. However, and despite these endeavours, the nature of some tasks and a lack of available experts inevitably results in some decisions being made by people who are less than expert.

(3) A subsequent objective was to select a specific area of decision making to develop a computerised decision support system. The research domain, recommending units for student selection, was chosen because it satisfied the criteria suggested by various authors; for example, the following responses could be made to eleven questions posed by Silverman (1987). The domain is an acknowledged problem area, an expert system may provide significant benefits, the administrators in the trial school had positive attitudes to technology, the problem has an identifiable solution, the problem's scope can be defined, the problem is not trivial, the problem is not suitable for implementation using conventional manual or computer systems, the problem occurs regularly, a solution will not be made redundant in the foreseeable future, there are experts available who are willing to spend time developing the system, and the solution could be relatively easily transported to other schools.

1.5 SIGNIFICANCE OF THE STUDY

Public schooling in Australia has, for most of this century, been characterised by highly centralised administrative systems. In the 1970s, some Australian States and Territories started to undergo radical changes and "a fundamental shift has occurred in the way Australian schools and school systems have been managed" (Beare 1989, 3). Beare also noted that "the trend around the world is towards deregulation of schools" (21). Reasons for these changes were summarised by Duignan and Macpherson (1991, 1) who cited a series of studies which identified three major trends: (1) an economic concern for efficiency, (2) the desire for educational effectiveness of systemic and school management, and (3) increased pressure for political effectiveness of school management. Nadebaum (1991) reasoned that these trends provided the opportunity to consider changing the way education is delivered in schools and thus to "better meet the needs of teaching and learning" (2). Whether by choice or direction, "the management roles in education are becoming both differentiated and quite diverse" (Beare 1989, 22) and "the kind of leadership needed now more than ever, must be knowledgeable as well as flexible" (Lakomski 1993, 2). However, Macpherson (1991), after examining the restructuring in Australia and New Zealand, drew attention to the "uneven level of technical expertise ... [as] ex-teachers had been promoted above and beyond the level and specificity of their expertise" (53). Telem (cited Hedberg *et al.* 1992, 134) identified the major problems faced by small educational organisations planning information systems as "the lack of specialised professional skills in system design and implementation and the highly fluid nature of the services and organisation structure." Hedberg *et al.* (1992, 138) aptly summarised the situation as follows.

> The need to examine the uses and potential uses of school information systems flows from the confluence of two current trends in educational administration. At the same time as the technology to manage information in the schools has become increasingly available, the need for more information and for more widespread use of the information has been tied to the efforts at making schools more effective.

Increased student retention rates, acknowledgment of student diversity and the need for flexibility in student choice, decentralisation of many functions including financial devolution and the increased responsibilities of school councils have each impacted on the nature of school administration in Northern Territory secondary schools. With this increasing complexity and range of personnel involved, there has been an expansion in the range of options available for many decisions. These decisions remain significantly reliant on the individual knowledge and experience of the decision makers, even though their knowledge and experience has normally been obtained in and for a different environment.

Clemson (1980) has noted that no computer application in education "has made a noticeable impact on the practices or the management of education" (98). Since then there have been many examples of computers impacting on the *practices* of education. There are even reports of expert systems that have been developed for student instruction or intelligent tutoring. However, Gould and Casperson (1991, 82) observed that "less [use] is made [of computers] in decision making in spite of the growing awareness within the administrative community of such computerised aids as decision support systems." Previous research (for example, Pratt 1983, Brown 1984, Crawford 1985, Hill 1986, Carbines 1987, Bucknall 1988, Hedberg *et al.* 1992) has examined the use of computers to improve the clerical efficiency and, at least indirectly, decision making in schools. The research presented in this thesis examines a means of directly assisting decision making in schools. The current research is significant as it appears to be the first formal study in Australia to examine the development of an expert system for use in school administration, and hence may contribute to and encourage future developments in this field.

The domain chosen for this study is significant because the subjects that students undertake "have a major influence upon the educational and career options available ... when they leave school" (Ainley *et al.* 1994, 13). Winning (1992, 41) reasoned that "the philosophical assumptions which lie behind curriculum policy come from particular historical, ideological and social contexts" and that the process of reviewing what knowledge is worthwhile will assist educational systems to "move beyond taking for granted the existing social dictates" (41). Education systems and schools thus need to ensure that the curriculum is appropriate and that the subjects available for students will satisfy systemic and individual requirements. The curriculum for Northern Territory schools is determined by the Northern Territory Board of Studies. "In Territory schools the curriculum consists of the total of all the planned learning experiences provided by the schools ... the high school curriculum attempts to ensure an appropriate degree of uniformity" (Northern Territory Board of Studies 1992, 4). The Northern Territory Board of Studies awards the Junior Secondary Studies Certificate to students on completion of Year Ten. To receive the certificate, students need to study the approved curriculum in ten subject areas "in accordance with minimum hours" (Northern Territory Board of Studies 1992, 4) for each of these subject areas. To facilitate the educational end-points, schools are involved in a variety of administrative and technical issues. Within the framework determined for schools in the Northern Territory, junior secondary students might exercise choice in subject selection but the choice offered to students by most schools is predominantly 'when' rather than 'what' subjects will be studied.

Recommending subject (unit) selection is a real problem, at least at the trial school, which has a unitised curriculum and a vertical timetable. Here the year-level curriculum set by the Northern Territory Board of Studies has been divided into discrete term-length units. Further, the school arranges classes using a vertical timetable to accommodate individual student progress rather than arranging classes by year level. Under these arrangements it is probable that a 'year nine' student may attend some classes which include 'year eight' and possibly 'year ten' students. Similar arrangements exist in many tertiary institutions but are less common in secondary schools.

Students are required to select their study units each term. Selection is based on the need to complete the minimum requirements, achievement in pre-requisite units, individual interest in particular subject areas, and on the need to prepare for further study and career aspirations. An effective computerised decision support system to recommend unit selection would help to ensure that student options were carefully examined and thus reduce the danger of wrong decisions. With the trend in increased litigation it could reasonably be argued that schools, and principals in particular, have a duty to ensure that they have not been professionally negligent in advice given to students and parents. The study is significant for the pilot school because the provision of a computerised decision support system to recommend unit selection should enable a collective expertise to be more readily available to a wider clientele. The principal, students, parents and staff could then be more confident that they have not overlooked important details, such as compulsory subjects or necessary pre-requisites for later units, when selecting study units.

Research by Martin and Law (1988, 580) indicates that the study is also significant for the trial school because the process of developing an expert system promotes discussion about the processes and reasoning and thus heightens the school communities' awareness of the issues and reasoning involved. A related benefit would be the proper documentation of this expertise to help safeguard the school against the loss of expertise arising from personnel turnover. Further, the resulting increased awareness within the school of the role of decision support systems may lead to other expert systems being initiated.

Gaines (1988, 271) aptly described the dilemma whether to cease introducing new technologies until users became competent in current systems, or to by-pass current technology and introduce new systems to overcome the current lag in training and competency. The research reported in this thesis is significant because it examines one attempt to by-pass current training and competency problems by replacing Management Information Systems with Decision Support Systems.

This research may also be viewed as a response to Zuboff's (1988) contention that organisations

> can choose to exploit the emergent informating (sic) capacity and explore the organisational innovations required to sustain and develop it (53). ... If not, we will be stranded in a new world with old solutions. We will suffer through the unintended consequences of change, because we have failed to understand this technology. By neglecting the unique informing capacity of advanced computer technology and ignoring the need for a new vision of work and organisation, we will have forfeited the dramatic ... benefits. Instead we will find ways to absorb the dysfunctions, putting out bush fires and patching wounds in a slow burning bewilderment (55).

1.6 STRUCTURE OF THE THESIS

This first chapter has defined the purpose of the research study, and its potential significance for the trial school and general research into expert systems.

The second chapter of this thesis reviews the literature, with particular reference to modelling, the development and use of expert systems, knowledge elicitation and representation, and the potential impact of an expert system. The chapter contributes to the thesis by presenting the argument that expert systems can be used to support decision making, and that computer systems are available to enable some schools to develop effective expert systems.

Chapter Three details the research aim, design, timing, and the research paradigm which has three main stages: modelling the domain, designing and implementing an expert system to recommend unit selection (named RUS from the domain acronym), and evaluation of this expert system.

Chapter Four describes the preliminary studies which were undertaken to select appropriate computer hardware and software, and determine a process for knowledge elicitation and representation.

Chapter Five describes the main features of the working system; the process of knowledge elicitation and knowledge representation, and the computer system used to model and present the knowledge in the expert system RUS.

An evaluation of the prototype is presented in Chapter Six. This chapter contributes to the thesis through an evaluation of the RUS system highlighting (1) the reliability of the expert system's recommendations compared to those of the human experts and (2) the positive response by users to the prototype expert system.

The findings are discussed in Chapter Seven, in which the success of the RUS system is used to support recommendations for the potential role of computerised decision support systems in the administration of schools.

The appendices consists of (1) the construction models used in preparing the unit recommendations, and (2) the programming code for the knowledge-bases.

1.7 CONCLUSION

This thesis (1) reports on an endeavour to design and implement a prototype expert system to demonstrate that expert systems have the potential to be relevant and useful in the administration of schools, and (2) suggests that schools can be significantly involved in introducing these computerised decision support systems.

This thesis is not intended to instruct readers how to develop an expert system; but it does aim to inspire readers to consider developing expert systems and highlights the key elements which need to be considered by schools before developing an expert system. The research does not aim to develop an expert system that can be universally applied; rather it reports on a longitudinal study in the development of an expert system for a specific domain in the pilot school.

Chapter 2

REVIEW OF RELATED LITERATURE

2.1 INTRODUCTION

Computer systems are increasingly being used to provide enhanced clerical efficiency in schools but are virtually unknown in school decision making. The emphasis remains on manipulating data rather than interpreting data. There is a number of possible reasons for this: a shortage of effective computer hardware and software, protection of current practices and status, fear of technology, and a lack of incentives and accountability.

Educators have tended to use computers as teaching aids and have started to use them as administrative clerical aids, but seem unaware of their potential use as management aids. That is, they may be using computers to enhance data storage, retrieval and presentation; but they have yet to use computers to assist decision making other than to provide the information on which to make decisions. This situation presents a contradiction in schools which aim to prepare students for the future, yet use the technology of the past. In light of the increased administrative and management complexity of schools, the result of financial devolution and other factors, computer competence will be essential if educationalists are to fulfil their dual role as administrators and educational leaders. Zuboff (1988, 55) stressed the importance of leadership that could exploit the capabilities of the new technology rather than be left "stranded in a new world with old solutions".

"We are beginning the transition from data processing to knowledge processing" (Feigenbaum 1989, 3), aptly described by Sell (1985, 1) as "from the tedious to the difficult". Decision makers may presently seek advice and support from human assistants, but in the future they are also likely to seek intelligent assistance from computers. There is emerging a new group of computer programs that capture the knowledge and experience of one or more human experts and which can mimic their capabilities and rule-of-thumb methods to solve appropriate problems. These computer expert systems may adopt a consultative process similar to that followed by the human expert: a request for information on a specific problem, the provision of a recommendation based on the knowledge available and the expertise of the system, and an explanation of the recommendation should this be required. Bielawski and Lewand (1988, 20) noted that human experts are not infallible and computerised expertise will also have weaknesses, especially since it is based on human expertise and as a consequence these computer systems should only be seen as tools to support decision makers.

The conspicuous benefits of developing an expert system are: (1) to convert intangible assets into information or real value that remains with the organisation, (2) to maintain an expert that does not have lapses and is able to give advice when other experts are not available, (3) to facilitate reliable problem solving which remains relevant through continual upgrading, and (4) to provide an expertise that can be replicated for distribution as well as provide a training tool for non experts. There are also indirect benefits that accrue because the process of developing an expert system is in itself an advantageous exercise for an organisation through the clarification of ideas previously taken for granted, the increased awareness of organisational structures, and the heightened discussion about the expert's knowledge and reasoning.

The use of models provides an opportunity to understand a real system by extracting and simplifying key features. Models can be used for description, to simulate or to explore alternatives. Because models are representations of larger complex systems, it is critical that appropriate entities and relationships are included but that non-essential elements are excluded. This selectivity needs to take cognisance of the modelling purpose and, equally important, other possible uses of the model need to recognise the limitations of the original construction.

The study of knowledge, an important precursor to expert systems, has a long and honourable tradition in philosophy. To avoid this philosophical debate, developers of expert systems tend to adopt a pragmatic approach and treat any rules, facts, truths, reasons and heuristics as knowledge if the experts have found these to be useful to solving problems in that domain. There are three traditional sources of this knowledge: literature, human experts and examples. These may be expressed as laws, experience and models; and may be perceptual, heuristic or strategic.

Knowledge Engineering is the general term applied to eliciting the facts and heuristic knowledge, encoding this information in the expert system's knowledge base and developing an inference system to use this knowledge. A central problem is not just in eliciting knowledge from the domain experts, but in eliciting the correct knowledge! Elicited knowledge may be encoded in a set of rules that cater for the varying degrees of confidence to reflect the uncertainties experienced in the real world. The inference process may proceed forward from the known to infer conclusions, or backwards from a given goal to detect solutions that will facilitate such goal. In support of the recommendations or conclusions, the knowledge engineering will provide the expert system with the capacity to explain the outcome. The capacity to explain what it is asking or recommending is central to the effective use of and confidence in an expert system, and is much more than a nice user-friendly touch.

The introduction of an expert system should adopt the same principles of any other change, but especially those which should apply to the introduction of new technology. Much has been written on the problems of introducing technological change. Problems inevitably arise when staff feel alienated by changes being thrust upon them. Successful implementation is more likely to occur when staffs are involved in the planning and implementing of change, when meaningful training and counselling are provided, when respect for individual staff and their skills are maintained, and when implementation is not rushed.

In addition to the particular issues already highlighted, a number of general concerns have also been raised: whether a computer can or should replace people; whether the advice can be trusted, and how it should be used and by whom; the impact on employee esteem and the impact on employment.

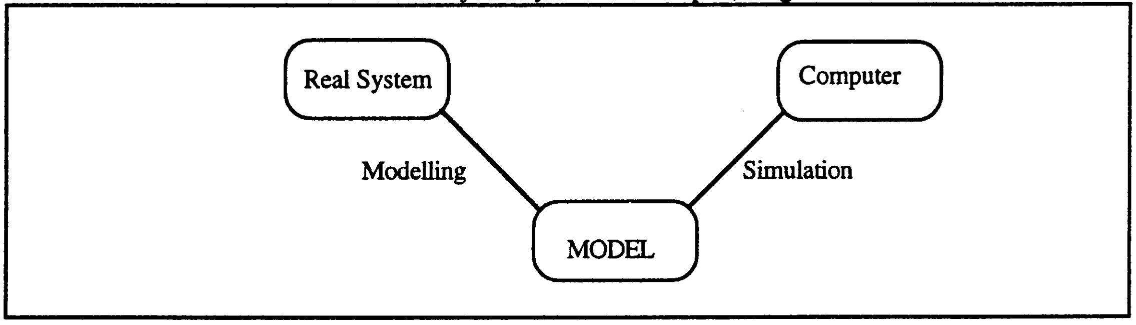
2.2 MODELS

A model can be constructed to represent theory, empirical observation or a combination of these (Fishman 1978, 2). There are many reasons for using models but typically they endeavour to represent the operation of an existing or proposed system. Effective modelling may result in an understanding of the real system, the opportunity to predict changes or the possibility of optimising performance (Osborne and Watts 1977, 6). The use of models may save costly, time consuming or even arduous real situations; models may be repeatable and non-destructive, and may provide statistical data for analysis (Zeigler 1985, 6). Many types of models have been used in the past but the development of computers has increased and will further propagate the applicability and practicality of models in old and new areas of application, especially because of the improved output presented via enhanced computer graphics.

One essential characteristic of a model is its incomplete representation of a real system, because only a careful selection of the real or proposed system's properties correspond to the model; though the actual model may be complete. Gordon (1978, 6) identified two main tasks in creating a model: establishing the model structure with its boundaries, entities, attributes and activities; and supplying the data to provide values for these attributes. The form of model needs to be related to the modelling purpose and thus it is important that the model is used with regard to its purpose and form.

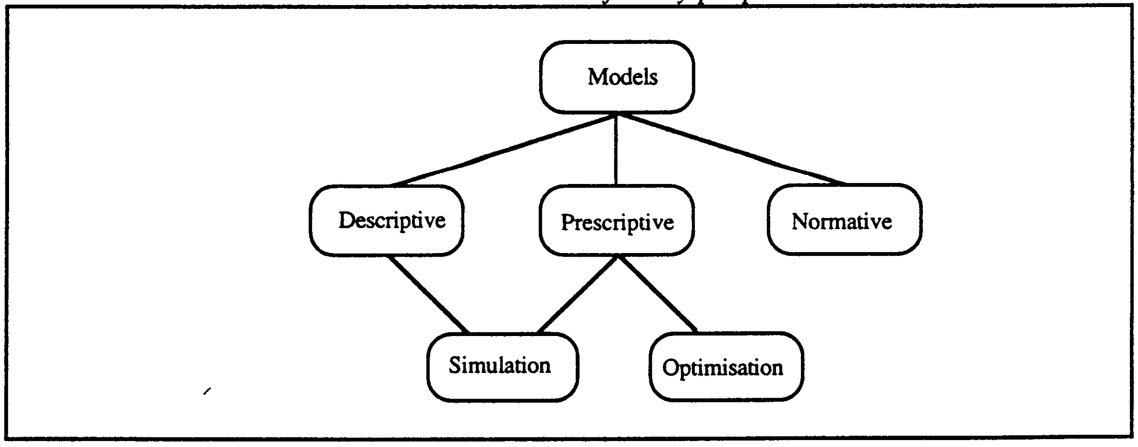
The differentiation between the development and use of models is illustrated in figure 2.2.1. In this relationship, modelling is the identification and representation of the real system whilst simulation is the process by which a model is manipulated to achieve the real system's essential characteristics.

Figure 2.2.1
Models classified by relationship (Zeigler 1985, 4)



Models can also be classified into categories analogous to their purpose, as shown in figure 2.2.2

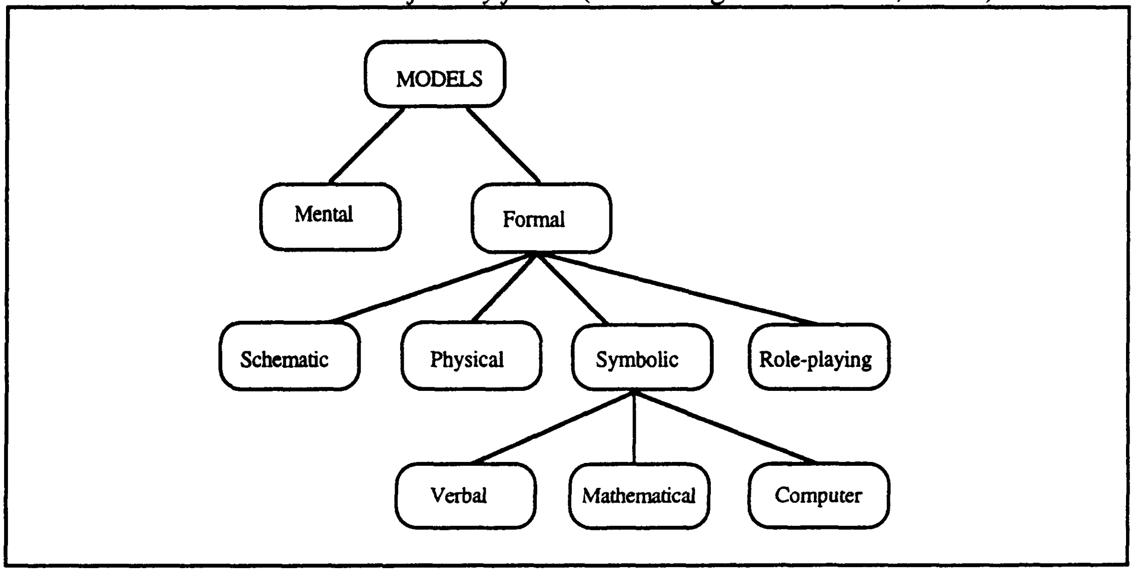
Figure 2.2.2
Models classified by purpose



Descriptive models provide an aid to understanding whereas prescriptive models aim to provide a solution and normative models describe goals and standards. Both descriptive and prescriptive models may use simulation to mimic the behaviour of the real system, prescriptive models may also use optimisation to solve problems by deriving the optimum from alternative solutions (Greenberger *et al.* 1976, 60). Hussey (1972, 19) noted that the modelling process usually begins with descriptive models before proceeding to quantitative models and the inferences which these may provide.

There is also a variety of model classifications based on the modelling language or mode of expression. For example, Gordon (1978) described a range of physical and mathematical models and Naylor (1979) examined corporate planning models. Other writers, such as Osborne and Watts (1977) and Ellison and Tunnicliffe Wilson (1984) have presented broader classifications of model forms which are compatible with figure 2.2.3.

Figure 2.2.3
Models classified by form (Greenberger *et al.* 1976, 50-52)



Mental models are subjective perceptions which need not be consistent but are frequently used by people in their daily response to situations and in lateral thinking, especially when the factors involved are neither numerous nor complicated. Formal models are necessary once the magnitude of a problem exceeds the capacity of the mental modeller, though formal models need not be excluded from simpler modelling exercises. Whereas mental models are elusive to scrutiny and modification, formal models are open to examination and disagreement and although not as immediately flexible may be modified and mutually accepted (Greenberger *et al.* 1976, 48). Littman examined the cognitive activity of building expert systems and concluded that "building an expert system is nothing more, or anything less, than building a mental model" (1989, 89) and that the computerised result is a translation of the designer's mental model to behave in such a manner that solves problems that experts solve — and are thus expert systems.

Because a model is intended to represent and simplify the reference system by including selected features and relationships, it is convenient to organise the system into a series of blocks based on the elementary entities and their immediate interactions. This extraction process facilitates concentration on a specific entity at a time, the blocks and their interactions can be independently tested, with further blocks being added as the need arises. In preparing the blocks, it is necessary to balance the need for detail with the need to simplify the reference system and avoid a potential combinatorial explosion of outcomes. Thus blocks will not necessarily represent all features of the reference system but will retain aspects relevant to the purpose and use of the model. Osborne and Watts (1977, 4) described this process as "simplification, idealisation and approximation". Greenberger *et al.* (1976, 63) detailed the need for balance between theory, data and methodology, which they reasoned were not independent of each other. A model based on sound and appropriate theory, using valid and appropriate data, and established methodology is not guaranteed of acceptance, but it is more likely to be accepted and used.

Fishman (1978, 3) remarked on three apprehensions apropos the use of models: (1) caution that time and effort do not guarantee success, (2) defence of the model by its developer may diminish its validity, and (3) the use of models beyond their original purpose. Hussey (1972, 19) also commented that the test of a good model inevitably depends not only on compliance with the prescriptions inherent in its construction, but also on its use which may be adequate for one purpose but not another and thus must be used with an understanding of its limitations.

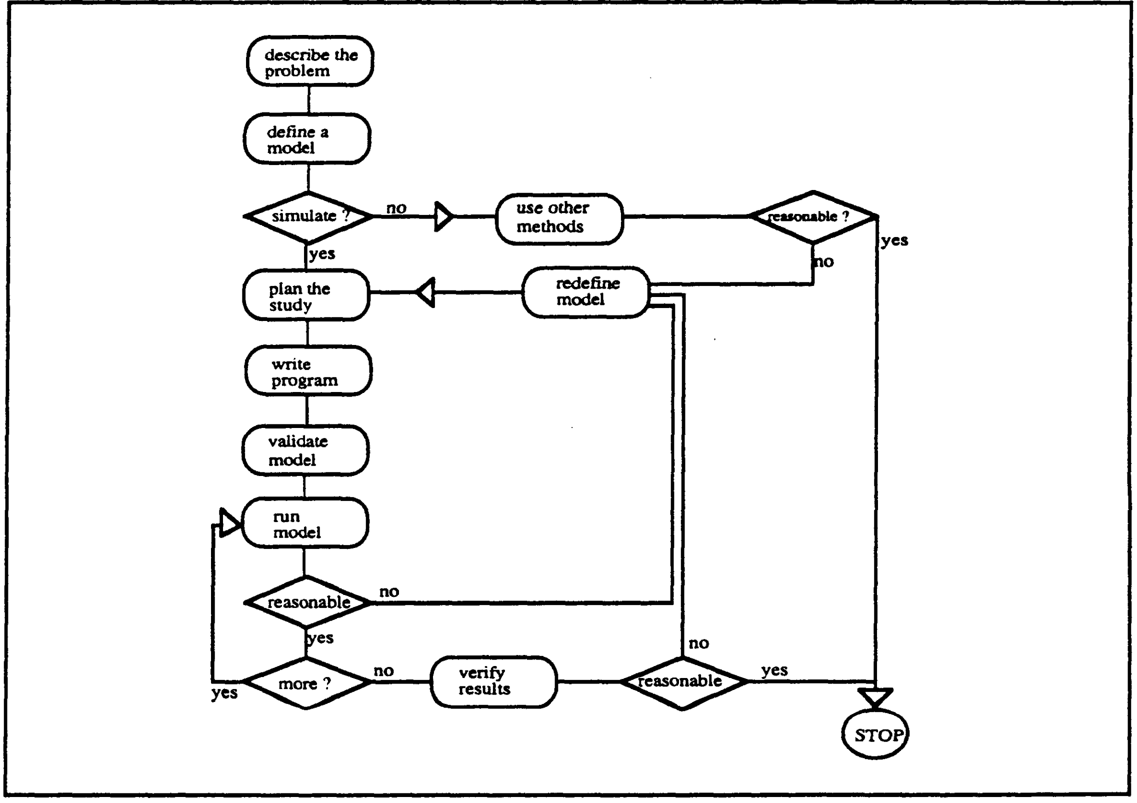
Models are simplifications of a real system and it is crucial that these abstract representations be tested during construction and on completion (Pidd 1989, 6). Osborne and Watts (1977, 23) outlined four causes of problems: incorrect methodology, poor experimental data, incorrect interpretation, and model instability. They also listed two steps in assessing models: comparison of the data input and output of the model and the real system; and prediction testing by comparing model output with real system experiments. Because assessment by clients and others depends on an understanding of the model, effective communication between designer and client is one of the most important aspects of modelling. Zeigler (1985, 7) advocated six steps to assist in making the model transparent and easier to assess:

- Informal description of the model and the assumptions that went into directing its construction.
- Formal description of the model structure.

- Presentation of the program with which the simulation was carried out.
- Presentation of the simulation experiments performed, and their results and analysis.
- Conclusions about the range of applications of the model, its validity and its running cost.
- Relating of the present model to other models.

Similar steps or procedures have been suggested by other authors, such as Alberts and Cormier (1988, 70), who described those models that cannot be professionally scrutinised as black boxes. Figure 2.2.4 is representative of flow charts which suggest an appropriate process for developing models.

Figure 2.2.4
Progress of a simulation study (Gordon 1978, 54)

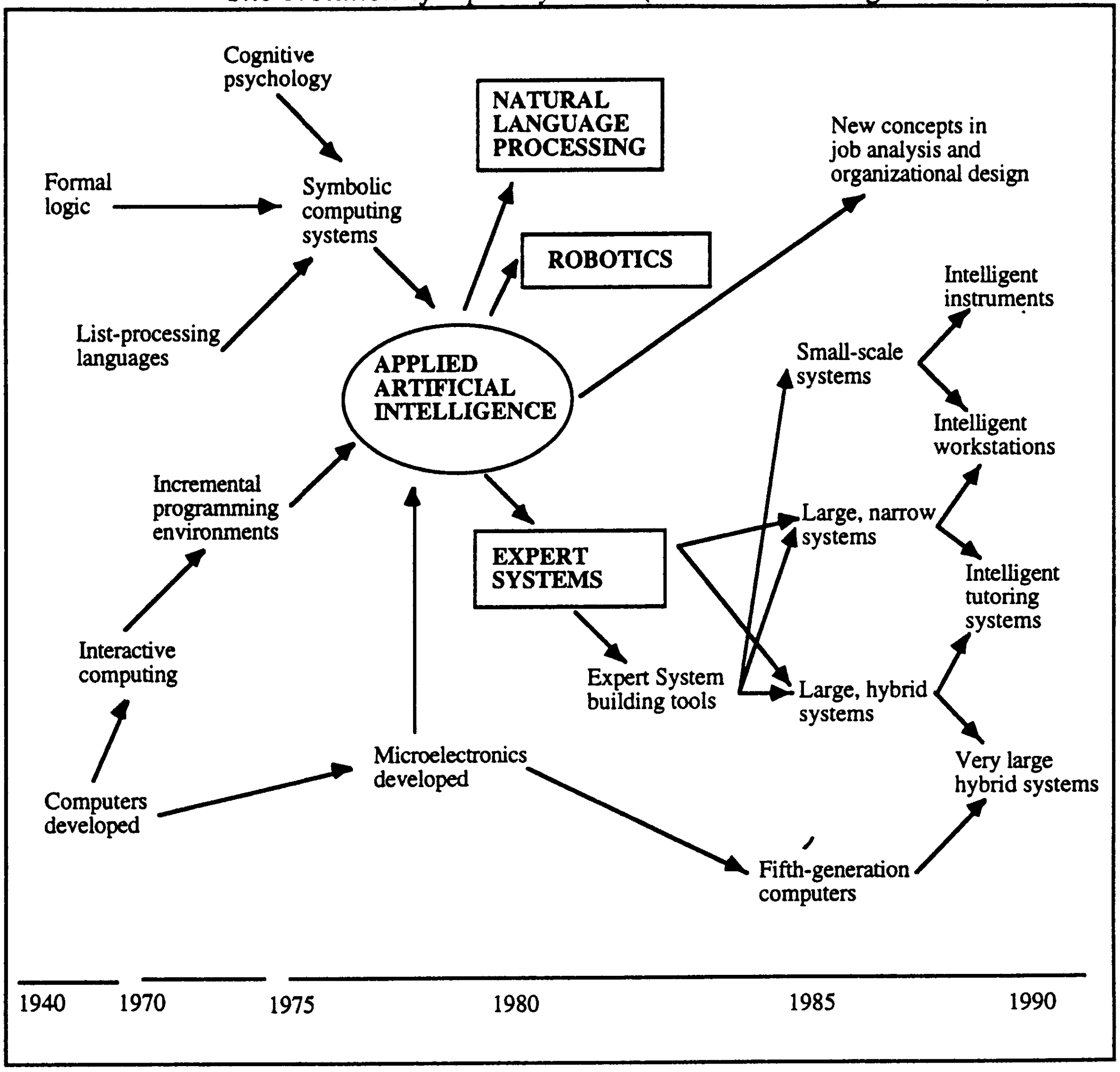


2.3 ARTIFICIAL INTELLIGENCE

Computers are relatively fast and reliable at processing specific instructions and thus computers have been increasingly used for complex mathematical calculations and filing information that could be sorted and retrieved. In addition to these tasks there have also been developments in knowledge processing ranging from manipulative file processing through to artificial intelligence. Increasing attention has been focused on the potential benefits of a natural language interface between users and computers as well as the desire to solve problems that are imprecise and thus beyond the capability of precise algorithmic processes. Barr and Feigenbaum (1981a, 3) noted that this move to develop computer systems that demonstrate reasoning characteristics normally associated with human behaviour will play an increasingly important part in the evolving role of computers in our lives to the extent that they will not merely be useful but essential.

Research during the late 1950s and early 1960s into artificial intelligence concentrated on discovering the fundamental laws of reasoning which a computer system would then be able to apply. Falk and Aungles (1987, 16) describe these endeavours as the strong claim in which it was asserted that artificial intelligence would eventually replicate human intelligence, as distinct from the more modest weak claim based on capturing a limited but useful sub-set of human skills. The failure of the early research into artificial intelligence to achieve satisfactory reasoning performance in the strong claim indicated that effective reasoning systems needed to include domain specific knowledge to at least provide some constraint to the combinational explosion that resulted during the earlier generate-and-test (trial-and-error) methods to find possible solutions to a problem. Further, many problems do not have algorithmic solutions as they are located in complex social contexts which resist precise description (Hayes-Roth *et al.* 1983, 4). Early commercial users of artificial intelligence were also often disappointed due to high product costs and/or low performance. As a consequence, a number of software producers went out of business and the term AI fell out of favour with many people (Light 1992, 134). Research into artificial intelligence has since fallen into three main subsets: robotics, natural language processing and expert systems (Knowledge Engineering), as illustrated in figure 2.3.1.

Figure 2.3.1
The evolution of expert systems (Harmon and King 1985, 3)


Cognitive psychology
Formal logic
Symbolic computing systems
List-processing languages
NATURAL LANGUAGE PROCESSING
ROBOTICS
New concepts in job analysis and organizational design
APPLIED ARTIFICIAL INTELLIGENCE
Intelligent instruments
Small-scale systems
Intelligent workstations
Incremental programming environments
Large, narrow systems
EXPERT SYSTEMS
Intelligent tutoring systems
Interactive computing
Expert System building tools
Large, hybrid systems
Microelectronics developed
Very large hybrid systems
Computers developed
Fifth-generation computers
1940
1970
1975
1980
1985
1990

2.3.1 Expert Systems

Much of the manipulative power once considered the domain of large mainframe computer systems is now available with mini-computer and also personal computer systems. These powerful tools have already enhanced productivity in many areas. A key element in this transition has often been the change in the information user/owner relationship typically changing from a 1:1 situation to many users sharing the same database. One result has been the need for increased attention to the problems related to the integrity and efficiency of large shared data-bases in an on-line environment (Garner 1987, 60). Also arising from the increased use of shared data bases by novice computer users has been the need for new interrogative processes more akin to natural languages.

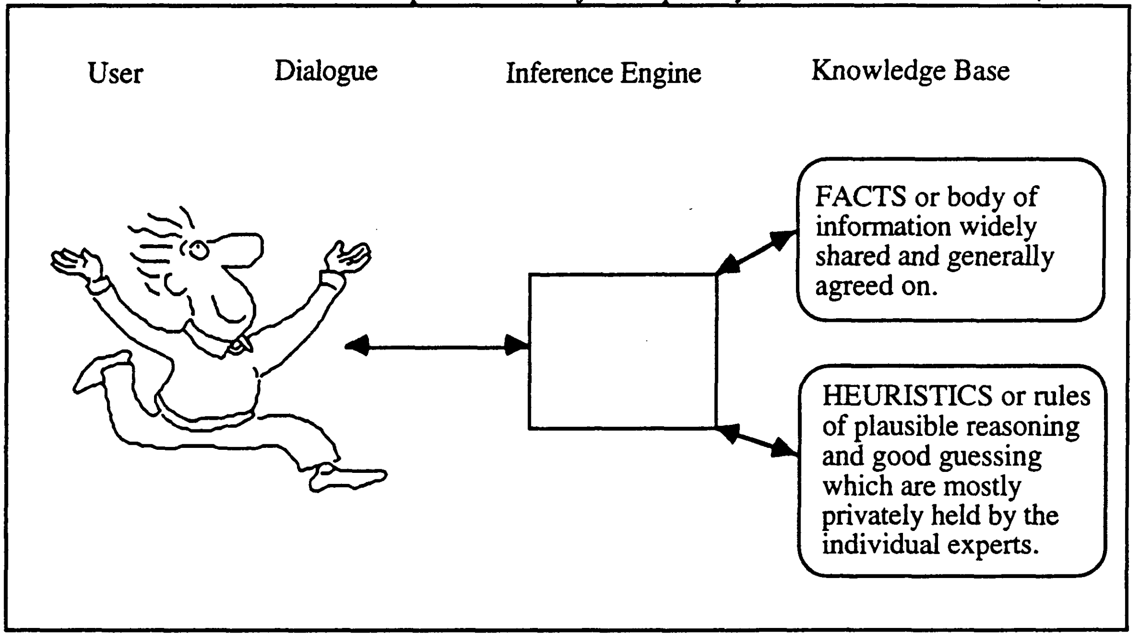
Hall (cited in Finlay 1986, 434) identified four phases associated with the application of administrative computer systems: (1) individual electronic data processing (EDP) of routine transactions; (2) linked electronic data processing systems sharing resources; (3) Management Information Systems (MIS) which provide information pertaining to current performance and/or information for planning; and (4) Decision Support Systems (DSS) with an emphasis on dealing with semi-structured problems. Expert systems, computer models containing both logic and knowledge data, are a sophisticated example of a decision support system and through their ability to explain their reasoning provide a distinct advance over earlier decision support systems. In practice, expert systems do not generally demonstrate intelligence, — that is, the ability to learn, respond to unpredicted questions, recognise their limits or make educated guesses when their data is inadequate. They can, however, deal with some problems beyond the capacity of conventional programming - especially problems that involve uncertain input and output derived from reasoning as well as facts.

The development of expert systems has tended to follow the same evolutionary path as other advances in computing. Davis (1985, 18) described this path as initially dominated by a group of "back-room *prima donnas*" using remote and awkward technology followed by improvements which, *inter alia*, allowed the technology to become more widely accessible, and which will culminate in personal expert systems becoming commonplace in society at large. A key factor has and will be the advances in computer hardware and software, especially in personal computers. Martorelli (1988, 56) noted that one of the largest expert system development markets has been in the United Kingdom where small personal computer expert systems have been actively advocated as ideal for exploration and experimentation by customers just beginning to work with expert systems technology. Analogously, Bielawski and Lewand (1988, 3) commented "just as the affordable Model T popularised the automobile. Inexpensive PC based expert system development tools are bringing artificial intelligence into the mainstream corporate America."

The main reason that expert systems have not become readily available yet is due to the complexities of expert systems development. Worthwhile prototypes can often be developed quickly, but development of a major operational system is a more complex and longer task. Although some of the problems stem from computing difficulties, the most often-cited bottleneck is the process of extracting and translating the expert's knowledge.

The aim of an expert system is to capture the specialised knowledge and experience of human experts, code this in a manner to be applied to a structured knowledge base pertinent to the given domain such that the computer simulates the conclusions of human experts confronted with the same problem. An expert system attempts to model the outcomes of a thought process rather than the thought processes themselves. Rather than apply strict application rules as in conventional programming, expert systems make use of heuristic rules that do not necessarily apply all of the time and thus are particularly useful in unstructured and poorly defined problems.

Figure 2.3.2
Descriptive model of an expert system


User
Dialogue
Inference Engine
Knowledge Base
FACTS or body of information widely shared and generally agreed on.
HEURISTICS or rules of plausible reasoning and good guessing which are mostly privately held by the individual experts.

Standard computer programs use exact instructions contained in an algorithm to manipulate a complete set of data to provide a unique solution to a given problem. On the other hand, people tend to symbolise their experiences and use knowledge that applies most of the time to solve many problems. The use of these incomplete sets of data may produce varying solutions to a problem, each with varying degrees of confidence, but which tend to work in a pragmatic environment. The core of an expert system is the inference engine; a computer program that applies domain knowledge to known facts in order to draw conclusions. Expert systems use relatively little coding and their power is found in the knowledge base, as distinct from conventional software which is procedural oriented and code intensive. The differences between conventional programs and expert systems are summarised in figure 2.3.3.

Figure 2.3.3
Some typical differences between conventional programs & expert systems
(Giarratano and Riley 1989, 47)

Characteristic	**Conventional**	**Expert System**
Control by	Statement order	Inference engine
Control and data	Implicit integration	Explicit separation
Control strength	Strong	Weak
Solution by	Algorithm	Rules and inference
Solution search	Small or none	Large
Problem solving	Algorithm is correct	Rules
Input	Assumed correct	Incomplete
Unexpected input	Difficult to deal with	Very responsive
Output	Always correct	Varies with problem
Explanation	None	Usually
Applications	Numeric, file and text	Symbolic reasoning
Execution	Generally sequential	Opportunistic rules
Program design	Structured design	Little or no design
Modifiability	Difficult	Reasonable
Expansion	Done in major jumps	Incremental

The increased sophistication of computer systems continues to break down the delineation between computer languages and the tasks they were initially designed to accommodate. Thus there are some programs which may be clearly identified as expert systems and others which clearly are not. For the end user there is little point in worrying about labels and ascertaining how many tenets of an expert system can be relaxed before the program is no longer considered an expert systems. Basden (1984, 71) and Silverman (1987, 12) noted that such distinctions are less important to users than the need to ensure that they obtain a system that will satisfy their requirements, which Hildreth (1989, 3) categorised as: an improved and wider useability plus a smarter functionality as part of the information retrieval system.

This may be placed in an historical perspective by observing that computers in the 1950s were often called electronic brains, even though they could do little more than arithmetic functions. Nowadays such machines are considered primitive and not at all intelligent. Similarly, the current work into expert systems will soon be considered commonplace and less than intelligent and the "mystery of intelligence will continue to be a moving target" (Myers 1986, 100).

There is some debate as to what constitutes an expert and thus the domains in which expert systems may be entitled to use the expert nomenclature. Savory (1988, 28) described an expert as someone who "has a large knowledge domain in the form of facts and rules, and in addition has individual experiences not found in the literature of the domain." Hayes-Roth *et al.* (1983, 13) described expertise as the difference between knowledge and skill. Joyce and Ramasamy Uthurusamy (1986, 1) also use the criteria of human ability to develop an expertise and to reason. They accepted, for example, that complex electronic diagnosis is an expert area, but they deny expertise to complex tasks which "many people could easily learn and purvey". Pedersen (1989b, 4) reasoned a wider view that "seemingly mundane tasks [such as scheduling] make excellent expert system targets", especially when the loss of knowledge accumulated by the present staff would involve lost production while other staff were trained. Carrico *et al.* (1989, 13) reasoned that the size of a problem should be "irrelevant in determining whether or not its worth tackling."

Over the last decade many expert systems have been developed and perhaps several hundred are currently in use over a wide range of domains: engineering and science, medicine and finance, for designing and planning, diagnosis and problem solving, monitoring and control, interpretation and training. The exact nature and extent of successful working systems is difficult to ascertain because many users are in highly competitive areas and are not eager to share their success. It is "often difficult to determine whether a system is in actual use, in the developmental stages, or a paper model only" (Scherer and White 1989, 290). Edwards (1991, 42) cited several surveys which suggest that although many expert systems have been developed, only a very small number are really in use, and Coats (1988, 403) cited estimates that less than one percent of serious expert system endeavours reach implementation. Munakata (1993, 3) summarised the history of expert systems in three eras: the academic era from 1965 to 1981, an expansion era from 1981 to about 1989, and the current massive production era. Lewinson (1994, 5) also described an optimistic outlook for expert systems, but acknowledged the "long [and] bumpy road for expert systems" and their failure to gain significant commercial success "in their first incarnation".

It appears that many organisations are at least considering expert systems. It has been estimated that there are from several to many billion dollars of current expenditure on expert system development in the United States of America (Coats 1988, 397). The indications are that expert systems in the United Kingdom predominantly use smaller personal computer based expert systems. The United States of America appears to be ahead in the development of larger expert systems, particularly in the problem solving and high technology sectors: telecommunications and electronics, defence and aerospace with less application in finance and manufacturing. Australia has adopted a wide use for expert systems, including advertising, finance, equipment maintenance and a wide range of planning and administrative tasks (Jones 1986, 115, Silverman 1987, 6, D'Angelo 1988, 56, de Jong 1988, 418, Plunkett 1988, 9, Bundy 1989, 42).

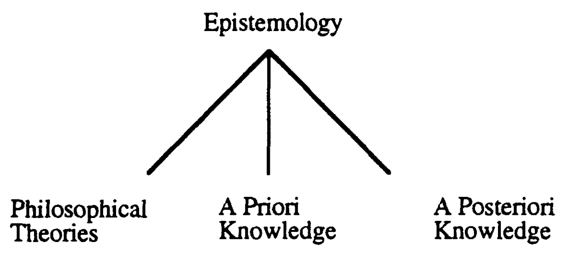
The 1985 report of the Australian Education Council Task Force on Education and Technology was prefaced as "the first national report to examine in a broad way both technology and education, with particular focus on their interaction" (Australian Science and Technology Council 1987). The report examined and made recommendations regarding the use of technology for the production of materials, as an instrument for teaching and learning, and for the delivery of education at a distance, but only briefly mentioned technology and educational administration, with a recommendation for improved communications within and between education systems. A dearth of examples for the use of technology in educational administration was also noticed during the relatively extensive review, reported in this chapter, of Australian and international literature. An on-line search of the Current Index to Journals in Education (CJIE), undertaken at the start of research for this thesis, failed to trace any citations linking 'educational administration' and 'expert system' and similar or related topics. A recent search indicated citations for educational administration and for expert systems but none combined. Checking each of the citations for expert system revealed articles on (1) a 1987 proposal to assist student admission at Northeastern University, (2) a 1992 prototype at the University of Wisconsin for students to maintain an individual academic advisory system, and (3) a 1992 project by an un-named College to advise students who are uncertain about college selection. When Bucknall (1993 and 1994a) published articles and addressed an international conference (1994b) pertaining to the research reported in this thesis, expressions of interest were received for further information but no comments indicating similar research undertaken elsewhere. Hedberg *et al.* (1992) commissioned an extensive review of American and Australian literature and surveyed schools in New South Wales, after which they noted that "schools have barely begun to utilise [decision support systems]" (138) and "that school administrators tend to use SIS [school information systems] for greater efficiency in traditional functions ... than for more effectiveness [in decision making]" (151).

2.4 KNOWLEDGE ENGINEERING

Even relatively simple tasks that require intelligent solutions are normally knowledge rich (Amarel 1984, 40). Because expert systems are data-oriented rather than procedure-oriented, it is necessary to establish and encode initial knowledge, reduce or avoid erroneous knowledge, and augment previously encoded knowledge.

Knowledge may be classified in a number of ways. Figure 2.4.1 illustrates general categories in the study of knowledge (epistemology) with particular distinction between *a priori* knowledge which is considered to be a universal truth that cannot be denied without contradiction, and *a posteriori* knowledge which can be denied without contradiction on the basis of new knowledge.

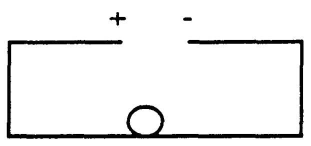
Figure 2.4.1
Some categories of knowledge (Giarratano and Riley 1989, 64) .



McGraw and Harbison-Briggs (1989, 22) detailed four further categories of knowledge: (1) procedural knowledge which involves a, usually reactionary, automatic response to a stimulus; (2) declarative knowledge, which can be expressed as facts; (3) semantic knowledge reflecting cognitive structure, organisation and representation, and thus tending to distinguish between people's level of expertise on a subject; (4) episodic knowledge, which may be described as perceptual characteristics. They concluded that it is necessary to use different techniques to elicit knowledge from these categories. Sternberg and Lasaga (1984, 220) stressed the importance of recognising that computer models aim to simulate human behaviour at a functional rather than a structural level. While the computer model may represent the steps taken by a human expert, it cannot claim to replicate these steps.

The general role of a knowledge engineer is to extract the expert's knowledge. It is generally accepted that this involves being critically involved from initial investigation through to on-going maintenance. One of the key functions is cognitive analysis, which involves eliciting and separating facts, rules, heuristics and inference strategies as illustrated in figure 2.4.2.

Figure 2.4.2
Illustration of and distinction between cognitive elements
(Adapted from Edwards 1991, 47)



Simple electrical circuit

Example of some Facts	Metals conduct electricity
	Copper is a metal
Example of a Rule	IF the connector is copper
	AND the battery is charged
	THEN a current will flow through the circuit
Example of a Heuristic	IF the bulb does not light
	THEN the battery might be flat (Probability 70%)
	OR the bulb might be faulty (Probability 20%)
	OR the connector may not be a conductor (Probability 10%)
	ACTION check the battery first
Inference Strategy	steps followed to solve problem by testing the possible solutions

Knowledge may be considered over three levels: rules, rules of thumb, and meta-rules. The rules provide a base of facts, theorems and operations. The heuristics, or rules of thumb, provide the first-order correction which are sometimes applied to the rules. Meta-rules provide second-order corrections to the heuristics. Waterman and Hayes-Roth (1983, 222) reasoned that if meta-knowledge is a factor then it should be recognised and dealt with explicitly. The use of demons is one way of achieving this. Demons are either internal or external functions which are not explicitly invoked, but are activated by a pre-defined condition. They are particularly useful because, once activated, a demon takes precedence over other processes until it is satisfied. Thus, for example, a demon activated within a program may result in that program being held in abeyance until an external condition is satisfied upon which the original program will resume. Alternatively, a demon can impose a condition which causes some rules to be fired and others denied.

There are four main technical aspects of knowledge engineering: (1) eliciting the knowledge, (2) representing the knowledge, (3) the reasoning approach to be used, and (4) the programming source.

2.4.1 KNOWLEDGE ELICITATION

In 1985, Sell (1985, 31) commented that "unfortunately there is no science of knowledge acquisition ... [and] what advice is available tends to be *ad hoc* and often no more than common sense". However, with the growing interest in expert systems more authors appear to be addressing this situation. Although Roth and Woods (1989, 245) stated that efforts to improve knowledge acquisition had been concentrated in either behavioural analysis research or the development of computer tools, the work by Cooke and McDonald (1986), Finin (1986), Brulé and Blount (1989), McGraw and Harbison-Briggs (1989), and others specifically addressed the pragmatic issue of system-oriented knowledge-acquisition methodology.

It is generally recognised that knowledge acquisition is probably the central problem in developing successful expert systems, a problem compounded by a mismatch between knowledge acquisition and representation (McGraw and Harbison-Briggs 1989, 18). A variety of reports suggest that at least six months and up to two years can be occupied in eliciting information, coding the data and completing even a small prototype.

Eliciting and formalising the expert's knowledge is usually a difficult task because an expert's knowledge is the sum total of different aspects of that person's entire life and even expertise within a particular domain may well be influenced by factors seemingly outside that domain. The expert's knowledge is often complex, unconscious and heuristic. Because the expertise is usually in solving a problem, rather than explaining all aspects of how the solution was derived, experts may simplify or forget some details and thus may not provide all the relevant relationships. In many situations people are simply unable to describe how they complete a task. Some facts and relations may be stripped of context and correlation when reduced to a set of rules; a process described by Bylander (1991, 74) as "knowledge approximation" rather than "knowledge compilation". Huang (1989, 489) summarised the situation thus: "most of these systems are brittle in the sense that they are not immune to even minor flaws in their encoded knowledge or slight changes in the environment".

An expert's knowledge may be used in three ways: (1) to develop a series of case studies against which similar problems can be compared, (2) by extracting a set of rules which can be applied to appropriate problems, and (3) by compiling causal models to simulate problems (Goel 1991, 72). However, the process of extracting a set of rules from explanations provided is unlikely to be completely successful, even after testing for inconsistencies. Williams (1986, 67) noted that these problems are further compounded by a general lack of skilled knowledge engineers to elicit and structure the experts' knowledge.

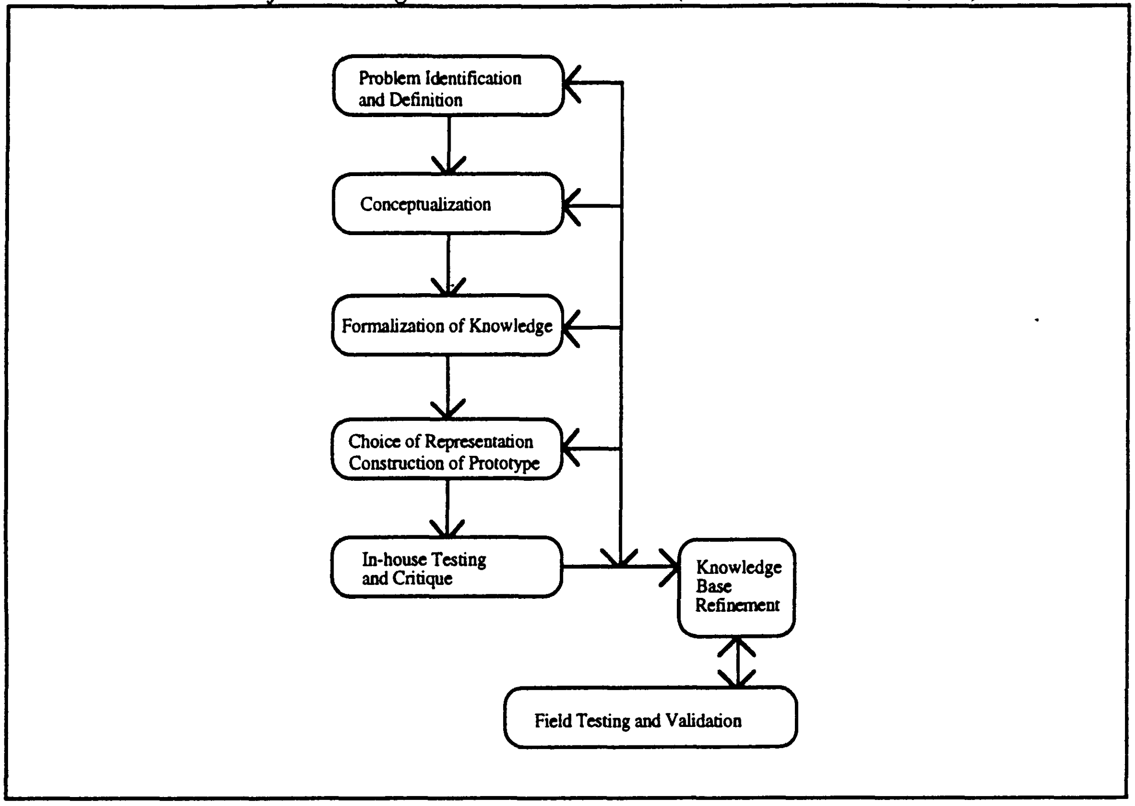
Knowledge engineers need to recognise that decision making and the process of deciding how to decide may be biased for a variety of reasons. Remus and Kottemann (1986) reviewed the literature pertaining to such bias and noted that bias in initial data may reduce the quality of decisions. This may be the result of excessive irrelevant data, the greater impact of human interaction over raw data, a preference for summarised data to raw data, the possibility of ignoring alternatives to the data presented, undue importance given to the first and last items presented, and the tendency to select data which conforms to expectations and previous experience. They also noted that bias in information processing may arise from the use of heuristics based on previous experience and the tendency to match new problems to old solutions and use old solutions which had been successful.

To reduce the cost of building and maintaining expert systems, a number of automated aids have been developed to manipulate knowledge elicitation, conceptualisation and representation. Although most of these are still experimental devices, some research has been undertaken to generate models, such as semantic networks, from which rules can be derived. Although usually developed as research prototypes, some at least have been used in exploratory applications. Kahn (1988, 30) and Buchanan *et al.* (1983) noted their weaknesses which include poor user interface, a tendency to meaningless or trivial generalisations, and abbreviated iterative processes. Thus far these tools have usually been used to support rather than replace the knowledge engineer. Systems include BLIP, DISCIPLE and LEAP (Kodratoff and Tecuci 1989, 136). Buchanan *et al.* (1983) outlined several systems, including EURISKO, META-DENDRAL and AQII. Marcus (1988) described five sets of tools for assisting knowledge acquisition, noting that each focussed on the particular problem solving technique used by its accompanying expert system. She cited six advantages in clearly understanding the role that domain knowledge plays in problem solving: (1) providing a focus for interrogating experts, (2) helping to identify missing knowledge, (3) assisting in determining appropriate use of the knowledge in the resulting expert system, (4) providing the source of end-user explanations, (5) help limit the task to its original purpose, and (6) assist mapping the domain expert's description of the problem. Thus these tools, although reducing the need for skilled knowledge engineers, still require an appropriate person to undertake an inventory of the domain and identify the symptoms associated with that domain. The tools also presumed identification of the problem-solving technique and types of relevant knowledge (Eshelman 1988, 47).

A collection of rules cannot replace a theory, and conclusions induced from an incomplete set of rules may be refuted when further facts and rules are ascertained. Thus, at the very least, provision must exist for knowledge base modification. Just as human learning is expanded and theories modified as additional information is presented, thus also an expert system needs provision for further input and modification. Ideally an expert system should be introspective about its knowledge and learn from experience. Kolodner (1984) has done some interesting research into reasoning models in which the program refines both the reasoning process and the domain knowledge. Huang (1989) proposed a framework in which feedback to the expert system, related to the usefulness or failure of the outcomes, from the actual situation (the environment) provided a basis for defining the usefulness of rules, which, in turn, modified their importance.

Sell (1985, 28) abstracted four stages in knowledge acquisition: converting it from an internal to external mode to make it available for examination, rendering it explicit to clarify the detail, recording it in symbolic form, and verifying the final symbolic model against the original intention. Sell's stages are compatible with other models, such as figure 2.4.3, which also reflects that, although the phases are theoretically sequential, work in one phase may affect earlier as well as later phases. Thus, in practice, these stages are "not clear-cut, well defined, or even independent" (Hayes-Roth *et al.* 1983, 24).

Figure 2.4.3
Phases of knowledge base construction (Kulikowski 1989, 164)


Problem Identification and Definition
Conceptualization
Formalization of Knowledge
Choice of Representation Construction of Prototype
In-house Testing and Critique
Knowledge Base Refinement
Field Testing and Validation

The data derived from knowledge elicitation techniques is usually in the form of an expert's verbal comments and actions. This may be obtained through interviews with, instructions by, or observations of, the expert; usually a combination of these approaches is used. This presumes that experts can be identified, are available, and are willing and able to share their expertise. Pollard and Crozier (1989, 50) reported on research into the validity of verbal reports and concluded that it was very unlikely that even experts perform tasks perfectly and that they are less likely to recognise influences that reduce their effectiveness. Pollard and Crozier recommended "a very strong cautionary note regarding asking people, experts or otherwise, to describe the basis of their decisions and judgments" (49). McGraw and Harbison-Briggs (1989, 17) cited research which suggested that experts may tend to see only one solution even though other effective solutions existed. Holtzman (1989, 46) further argued that, since "most people find it unproductive to be their own devil's advocate", the decision analyst and decision maker should be separated.

Interviews require little equipment, are "highly flexible, portable and can yield a considerable amount of information" (McGraw and Harbison-Briggs 1989, 9), though the ratio of relevant to irrelevant knowledge can be very low (Bachant 1988, 202). Further, interviews typically involve lengthy sessions in which the knowledge engineer endeavours to establish facts and rules for the domain. Protocol analysis involves observation for patterns, heuristics and other clues as an expert solves a problem. Scaling techniques use descriptive statistical techniques to represent relationships. These discoveries need to be skilfully interpreted to ascertain the underlying knowledge and to discern that "people are better at remembering events or facts when provided with clues" and thus structured tasks should be more successful when eliciting ideas from experts (Cooke 1989, 61). Morik (1989, 131) concluded that modelling is an interactive process which reveals laws, but through a process requiring reversibility and the possibility of starting all over again because of their tentative nature.

When developing an expert system it is necessary to achieve a fine balance between a narrow, implementable domain that is yet wide enough to be useful (Sell 1985, 16) but avoiding the danger of becoming "bogged down in a combinatorial explosion of preferences" (Eshelman 1988, 79). An ever present danger is that information may be ignored because it does not appear to be relevant or important. Thus it is recommended that "a rich view of expert knowledge should be adopted" (Kidd 1985, 16). However, Holsapple *et al.* (1987, 290) noted that rule syntax can lead to fragmentation of a conclusion into multiple rules and this proliferation can impinge not only on the rule specification process but also on inference efficiency. Further fragmentation can occur when the use of several experts results in diverse or conflicting information. This problem may be further compounded as experts sometimes contradict themselves when considering their own stated knowledge from a different viewpoint at a later date (Ramsey and Basili 1988, 45). The reality is that most complex problems require access to a number of specialists even when there is one expert who may be able to cover these fields, if only because THE expert is unlikely to have sufficient time to be available whenever needed, and there are potential benefits in eliciting more than one solution to a problem.

There are three typical knowledge sources for creating a knowledge base: human expertise, textbooks and manuals, and examples from previous cases. Although the first is not always easy to extract, and the second is not always available, it is perhaps surprising that greater use is not made of examples of previously solved cases (Kulikowski 1989, 165). Although considerable effort may be needed to find and collate such archival material, such data may be invaluable not only in preparing but also testing an expert system.

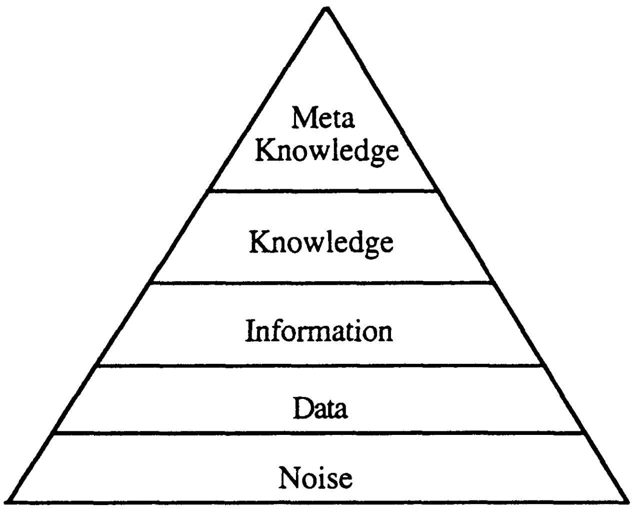
The development of an inductive knowledge base is an alternative to manual rule creation. Here the expert provides examples of problems and their solution and it is the function of the expert system to induce the rules to be used in conjunction with the knowledge base. However, common to other knowledge elicitation techniques, information may be left out or denied because it did not seem to be important at the time or was simply overlooked; thus the induction tool may build its conclusions on incomplete and potentially misleading information. A number of expert systems developed using the iterative refinement approach have failed when too few cases have been used to cover all the possible outcomes, or developers have bowed to pressure to put the system to work sooner rather than later. Further, examples will not necessarily provide the structure or mechanisms to cater for new and different situations (Roth and Woods 1989, 235).

Thus far, inductive systems have only provided limited flexibility because few useful general principles have been articulated and there do not appear to be many situations where concept learning has been successful in creating or enhancing expert systems outside the laboratory (Whittaker *et al.* 1989, 18). Sell (1985, 29) noted that induction methods worked best in scientific disciplines where the heuristics are less partial; similarly, Bielawski and Lewand (1988, 14) suggested that an inductive system is more appropriate if the information is derived from a spreadsheet or database which already has distinct relationships.

2.4.2 KNOWLEDGE REPRESENTATION

Whereas conventional programs are based on data structures and fixed algorithms, expert systems are based on inferring information and knowledge from data. Figure 2.4.4 helps to illustrate this hierarchy of knowledge. Analysis of the knowledge implies that it can be decomposed through cognitive and evaluative processes to reveal key components and their relationships. This analysis may also indicate other knowledge required for the expert system to function properly and/or the need for alternative knowledge elicitation techniques.

Figure 2.4.4
A hierarchy of knowledge (Giarratano and Riley 1989, 65)



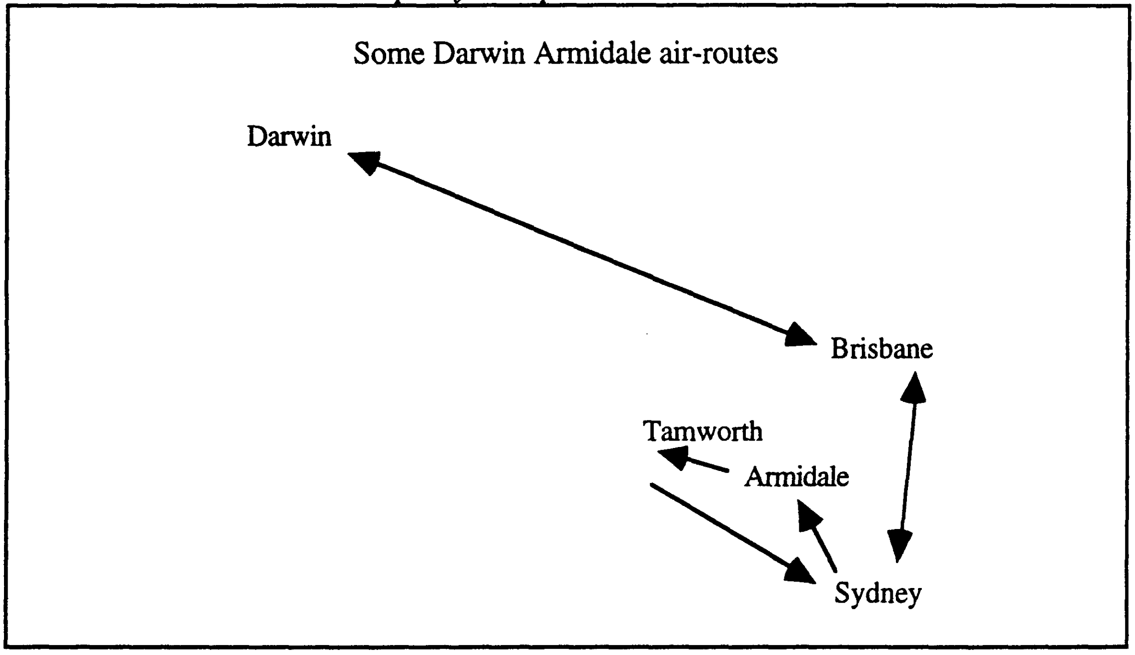
The first task, which also continues throughout an expert system project, is to analyse the elicited knowledge and abstract the general concepts and heuristics to ascertain the overall patterns between concepts and attributes. Once these patterns have been established, the individual data structure may be represented in a variety of formats to construct a model of how the human expert's mental processes are organised. This is normally a two step process: knowledge analysis using techniques such as semantic networks, decision tables and decision trees to provide a formal model of the domain; and knowledge coding using techniques such as rules and frames to transform the model into working code. Some literature makes the distinction between model based and rule based problem solving, but Chandrasekaran (1991, 75) reasoned that these are not "genuine alternatives ... [as] all descriptions of any aspect of reality are models" and that "the important research issues have to do with how problem solvers can move flexibly from one model to another in the pursuit of goals" (79). Baskin and Michalski (1989, 113) also reasoned that there was no "single best" paradigm for representing knowledge or problem solving.

The relationships established by an effective format are critical to the success of an expert system. Gevarter (1982, 20) cited the game of chess as an example of incremental pattern based networks in which thirty rules could be used to cover approximately two million configurations in a King and Knight verses King and Rook situation. The power of these relationships is a key to distinguishing between expert systems and conventional programming. The knowledge analysis and coding also need to deal with potentially large search spaces, and thus need to be factored or pruned to enable effective examination.

SEMANTIC NETWORK

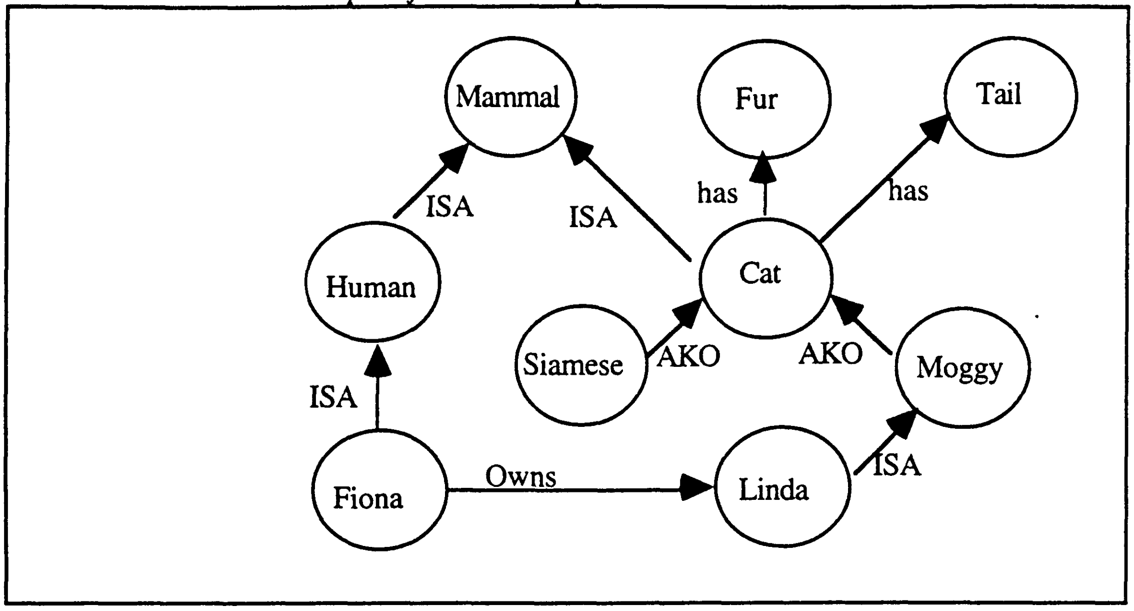
These networks help to identify relationships among a set of objects which may have subordinate relationships and property characteristics. A simple semantic network may be seen in airline route maps such as figure 2.4.5

Figure 2.4.5
Example of a simple semantic network



The structure of semantic networks is based on nodes (objects) connected by arcs (links or edges). The arcs are the critical component converting the collection of unrelated facts into organised knowledge. ISA (is-a) and AKO (a-kind-of) are two commonly used arcs to demonstrate relationships and characteristics. For example, from figure 2.4.6 it can be inferred that Fiona owns a cat with a Tail.

Figure 2.4.6
Example of a more complex semantic network



Semantic networks have a simple syntax and are very flexible, and thus easy to modify. This flexibility may be of benefit in enabling a variety of models to represent a situation, but may be disadvantageous if links and nodes become tangled through excessive intertwining. To reduce this problem, one approach to maintaining order encourages a hierarchical approach within the semantic network (Warnier/Orr cited in Carrico *et al.* 1989, 65). Giarratano and Riley (1989, 80) noted that semantic networks are useful in showing binary relationships but that searching nodes can lead to a combinatorial explosion, especially when the response is negative. Further, such nets are "logically inadequate because they cannot define knowledge in the way that logic can".

DECISION TABLE

One problem in keeping track of knowledge is to ensure that redundant and contradictory details are excluded from the knowledge base. A strategy, advocated by Francioni and Kandel (1988), is the use of decision tables. Figure 2.4.7 illustrates a decision table to identify fruit. However, although decision tables may provide an effective way of capturing, cataloguing and sorting data, they do not clearly describe a decision process (Carrico *et al.* 1989, 74).

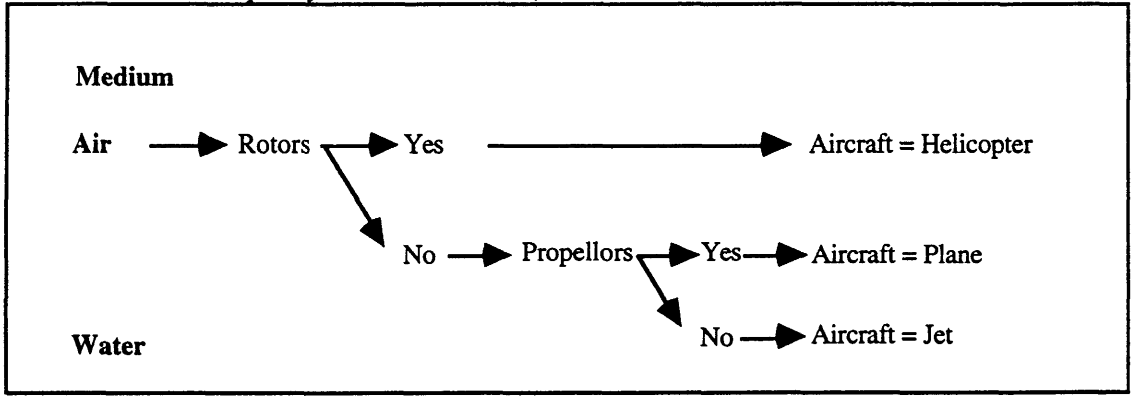
Figure 2.4.7
Example of a decision table (adapted from Carrico *et al.* 1989, 75)

Attributes				
shape	round	round	oblong	oblong
smell	acid	sweet	sweet	sweet
colour	yellow	red	yellow	green
taste	sour	sweet	sweet	sweet
skin	rough	smooth	smooth	smooth
seeds	yes	yes	no	yes
Conclusions				
grapefruit	Y			
apple		Y		
banana			Y	
pear				Y

DECISION TREE

A decision tree pictorially represents the relationship between items in a certain class, as illustrated by figure 2.4.8 and is a relatively easy way to represent complex diagnostic structures.

Figure 2.4.8
Example of a decision tree (Bielawski and Lewand 1988, 25)



Tree searching has been used previously in mathematical optimisation and is thus not unique to the field of artificial intelligence; but what is unique is the use of heuristics to ascertain the strategy for searching the tree, based on the implications developed by the expert system (Keller 1987, 98). For example, by translating the decision tree into production rules then such rules can be used to determine the line of questioning. In figure 2.4.8 the process could be along the lines:

If question = "Does it have Rotors"
and response = No
then question "Does it have Propellers?"

```
If question = "Does it have Rotors"
and response = Yes
then answer "Aircraft is Helicopter"
```

However, these structures suffer the disadvantage of drawing users in a forward direction and thus are not applicable in situations when a backward reasoning approach would be more appropriate. Further, noted Holtzman (1989, 57), they usually "do not allow independent relations to be exploited".

RULE

In a rule based system the expert's advice is expressed as a series of IF-THEN statements. Each entry in the knowledge base is a rule of the form

IF antecedent *THEN consequent.*

If certain attributes have certain values then other attributes are asserted to have certain values. This relatively simple format has two significant extensions: (1) the antecedent may have conjunctions [this *and* that] as well as disjunctions [this *or* that], while the consequent may only have conjunctions; and (2) both the antecedent and consequent may include factors which are less than fully true. The main advantages of rules are that (1) their order is not critical and they can be modularised and grouped together for easier comprehension; (2) individual rules can be written in language that users can understand; (3) more than one rule can contribute to the conclusion; and (4) individual rules can be added, modified, or deleted. The main disadvantage of rules are that (1) some representations are unwieldy as there is almost no capacity for generalisation, or impossible because they lead to circular reasoning, and (2) the format of production rules is difficult to apply in diagnostic problems.

Rules are transparent because their contents and meaning stand alone without relying on their location or order. However, this location flexibility can also result in rules having a low visibility which may present difficulties when evaluating the knowledge base. Access to rule tracing facilities may be essential, especially in systems with many rules that cause the firing of other rules. Jackson (1990, 152) noted that rules are good for linking conditions with actions, but are less suitable for representing knowledge about events or objects.

Figure 2.4.9
Example of a rule base

IF	Temperature is over 100 OR
	Pressure is over 250
THEN	Situation is dangerous
IF	Time is after 1800 OR
	Time is before 600
THEN	Hot water not required
IF	Situation is dangerous OR
	Hot water not required
THEN	Turn off boiler

In response to an inquiry, the antecedent of each rule will be examined and, if satisfied, the consequent will be enacted. The action resulting from the firing of a rule may call on other rules. The rule base will continue to be examined until no additional rules are fired. Not every rule will necessarily be fired despite many passes through the rules. For example, rules about birds are unlikely to be fired for an inquiry about four legged animals. Some rules might only be fired as the consequence of other rules firing. For example, rules which distinguish between breeds of dogs would not be fired until another rule identified the four legged animal as canine. Data may prevent some rules from being fired. For example, the four legged canine may be male and thus female characteristics may be irrelevant.

FRAME

Frame systems are useful in the classification of knowledge by using a representation which is true for the majority of cases in that classification, for example, that birds fly. Things and events are represented by a collection of frames in which there will be one entity per frame but with labelled slots for other information pertinent to that frame. Thus it is through the existence, or non existence, of these slots that information is related or excluded. Figure 2.4.10 illustrates a frame for Mammal which contains the default values to save repetition in the other frames. The other frames contain specific values for individual mammals, including exceptions to the default values. This feature may, however, be a problem if every frame is able to deny inheritance factors thus detracting from the reliability of the composite frame (Giarratano and Riley 1989, 86).

Figure 2.4.10
Example of a frame knowledge base

MAMMAL	
Skin	Fur
Birth	Live
Legs	Four
Infant Food	Milk

MONKEY	
A KIND OF	MAMMAL
Tail	Curly
Legs	Two

RABBIT	
A KIND OF	MAMMAL
Ears	Long
Moves	Hops

WHALE	
A KIND OF	MAMMAL
Moves	Swims
Legs	None

Frames can be put together in sequence, linked hierarchically or in a network such that a frame may inherit information from more than one ancestor and thus are potentially useful for large data structures which can be naturally organised into relationships. Frames can include both declarative and procedural knowledge (Alty 1989, 190). Jackson (cited in Giarratano and Riley 1989, 85) noted that "the frame paradigm has an intuitive appeal because their organized representation of knowledge is generally easier to understand than logic, or production systems with many rules". Because frames are based on hierarchy and inheritance, they are useful in representing knowledge involving cause and effect, whereas rules may represent unorganised knowledge that does not have a causal relationship.

One study (Ramsey *et al.* 1988, 46), using two systems on the same problem (involving a new field with unclear knowledge), demonstrated that the rule system "provided more interpretations and exhibited a higher rate of agreement with the database than did the frame system. ... However, as a field becomes more established, a frame-based system may provide better solutions". Brulé and Blount (1989, 123) stressed the benefits of adopting a flexible approach when selecting rules and/or frames to ensure that knowledge was not lost. Frames are often used in conjunction with other systems, such as rules. Examples of this combination include LOOPS and KEE where the frames provide a rich data structure for objects referred to by the rules (Srihari 1989, 6).

UNCERTAINTY

A potential problem with rules relying on antecedents and consequences is the difficulty created by competing hypothesis; that is, in situations when the input is not conclusive and/or when the domain knowledge is imperfect. There are a variety of methods for dealing with unreliable data and knowledge. For example, probabilistic reasoning (Bayes), Dempster-Shafer theory, set-covering techniques, fuzzy sets (Zadeh) and certainty factors (MYCIN). The use of certainty factors is an increasingly commonly adopted departure from predicate calculus

Heuristics are the general rules-of-thumb which people often apply in formulating a decision. Just as these are not always true in real life, so the expert system needs to make allowance for the degree of certainty or confidence that it can have in a specific rule. Thus the knowledge bases of many expert systems need to cater for the confidence of rules through a spectrum of certainty factors (CF), as illustrated in figure 2.4.11, so that structure of rules changes from 'IF this THEN that' to 'IF this (to some extent) THEN that (to some extent)'.

Figure 2.4.11
Example of certainty factors

1.0	absolutely right
0.75	probably true
0.50	reasonably correct
0.25	possibly true
0.0	unknown
-0.25	possible false
-0.50	reasonably false
-0.75	probably false
-1.00	absolutely false

These certainty factors will be used by the inference engine in determining whether a condition has been satisfied. Thus, for example, a rule with a certainty factor of 0.25 might not be deemed to have satisfied a condition and thus not held to be true. On the other hand, a combination of certainty factors may be held to satisfy that condition. In some programs, the certainty factors for a given situation might not necessarily add up to 1.0; for example, the probability of success could be 0.9 even though the probability of failure may be 0.3. Not all software is this flexible, however!

Jackson (1990, 99) noted that although there is general agreement amongst researchers that confidence factors are important, there are diverging views on their method of calculation. The difficulty in satisfying statistically correct probabilistic approaches, such as Bayes's theorem (detailed in Negoita 1985), has resulted in greater acceptance of subjectivist probability involving human judgement. Certainty factors are therefore often *ad hoc* just as expert reasoning is also often *ad hoc*; but the important thing in this situation is not their pedigree but whether they work. Silverman (1987, 17) described this approach as “phenomenological”, an approach which is used because it works, despite its lack of rigorous theory. The important thing to remember is that the knowledge content of the rules is more important than the algebraic confidences that hold the system together (Luger and Stubblefield 1989, 311).

The utility of certainty factors is repudiated by some researchers, such as Holtzman (1989, 95), who argue that attempts to use certainty factors have failed because of variations in individual assessments of probability, because the range of outcomes is unimaginably large, and because they are of limited practical use in decision making. A further disadvantage is that a string of consequential rules will gradually reduce the overall confidence in the result, an outcome that might not reflect the expert's thinking. Some systems accommodate this by, for example, only applying the highest certainty factor of contributing rules.

The uncertainty concept is further extended through the use of "fuzzy logic" which provides a link between the numeric needs of the computer and the imprecise facts of the real world. The use of quasi natural language allows, for example, age to be expressed by the terms young or old. The uncertainty of such an answer may arise because the user does not know the person's age; or it may result from uncertainty on how to define age. An Australian thirty year old is not usually considered middle-aged, nor is someone who is thirty years and one month. But how many months after thirty years is one middle-aged? The use of fuzzy logic, as illustrated in figure 2.4.12, helps to resolve this problem and thus would appear to be well suited to some expert system environments. Fuzzy logic may also be seen as an extension of certainty factors in ascertaining membership of a given set when a datum may only exhibit some of the properties associated with that set. Fuzzy logic is to meaning what Certainty Factors are to confidence.

Figure 2.4.12
Example of a fuzzy logic set (Adapted from Summers 1990, 292)

Ascertaining middle-age

Age	Strength of set membership
30	not a member
35	membership uncertain
40	probably a member
45	full member
55	full member
60	probably a member
65	membership uncertain
70	not a member

Although the key ideas for fuzzy set theory were first espoused by Zadeh in 1965, the theoretical concepts have been developed most over the last two decades. Klir noted that the theory continues to advance rapidly in sectors of the academic community, but that applications and wider interest appear limited other than in Japan where there were "some highly successful applications of fuzzy control in the late 1980s" (1991, 8). Munakata (1993, 4) reported on the more recent use of fuzzy logic systems in a widespread and expanding range of electrical appliances.

2.4.3 PROGRAM REASONING

A chain of reasoning is formed when multiple references connect a problem with its solution. Forward chaining is the processing strategy in which the program commences with the known facts and attempts to infer conclusions from these. Forward chaining is useful when the user needs to know everything about the domain which applies to a given situation. Backward chaining commences with the final goal and seeks information to solve that problem, but in the process ignoring the implications of facts that do not appear to have immediate significance. An example of forward chaining would include a student wishing to know the subjects available given current achievement, whereas backward chaining would indicate the subjects required for entry to a given course. Although some computer software accommodates forward and backward chaining, most expert systems only chain reasoning in one direction. The reasoning approach to be used will have implications for the knowledge engineer in the design of the knowledge base and selection of software.

Figure 2.4.13
Chaining characteristics (Based on Pedersen 1989b, 80)

Trait	Forward Chaining	Backward Chaining
solutions are	not prenumerated	prenumerated
the goal is	not necessarily known	known
the objective is to	flush out all facts	infer key facts
starting situation	some facts known	few facts known
questions and answers		
user input required	not necessarily	yes
% rules typically applying	relatively high	relatively low
strategy	build solution	detect solution
problem type	configuration	classification
	planning	selection
	interpretation	diagnostic

REASONING INTERFACE

The quality of reasoning is a critical aspect of an expert system. But even "excellent decision making performance does not guarantee user acceptance" (Langlutz and Shortliffe 1984, 77). If people cannot question and follow the reasoning process of an automated system, they cannot be expected to endorse its decisions and hence may not use it even if the computer system is known statistically to outperform the user on that particular task. Without an effective interface between the user and the expert system, the user might override the expert system's recommendation through ignorance of how it was reached. Preparing an effective interface is a valid and important aspect of the knowledge engineer's job. Construction of the interface needs to be considered in conjunction with the knowledge representation and computer software.

In addition to preparing an interface able to meaningfully explain the reasons for requesting particular input data and to provide justification for any recommendations made, the knowledge engineer needs to provide audit facilities to enable queries over the knowledge base procedure or accuracy to be checked. A range of audit facilities, to facilitate confidence in advice given by expert systems, is examined in section 2.6.

The early expert systems, in providing any explanation, tended to quote a rule. Such explanations were, at best, only a by-product of the system's ability to trace the rules which had been fired (Brachman *et al.* 1983, 48). Later systems increasingly include provision for text within the rule structure, and/or for plain English rules, so that users can understand them better. Some systems include specific text segments for user-interface while allowing mnemonics for programming. Slagle (1988, 93) described the development of an expert system shell AGNESS which provided eighteen explanation types: six forms of response (who, what, when, where, why, how) and three categories (past, present, future). But these are still short of providing explanations in terms of basic principles and a rational line of argument using an unlimited vocabulary and syntax capable of reasonable human discourse, and well short of speech synthesis and recognition.

It is important to avoid frustration that would be created by providing a technically correct response when the nature of the feedback was not what the user expected, especially if a null answer was provided. Most attempts to deal with this situation have used natural language interfaces. Some have adopted a loose structure data model using an object-oriented logic based approach to at least distinguish between user mistakes and genuine null responses, so that users do not presume the null response was an error on their part and needlessly try again. Another approach is to present users with a complete menu of questions that can be put to the expert system and for which a valid response will be provided (Olson and Lindahl 1988, 82). Shin (1988) described preliminary results of the ongoing project MIDMAN which intended to accommodate incomplete user queries through a natural language interface.

Systems which provide facilities for users to ask questions increase the range of situations that the system is expected to cover. This situation obviously becomes even more complex when responses are open ended and extended by variables. Motro (1986, 598) cited several cooperative systems to interpret failures and improve the interaction by monitoring the data and informing the user when a change occurs for example, "*No you are not eligible to enrol in that unit - shall I let you know what you need to complete first*".

Langlutz and Shortliffe (1984, 77) outlined a program (ONCOCIN) which went one step further to provide a critique on alternative solutions provided by human experts. Wick and Slagle (1989) designed an expert system in which the user interface included an *Interface Engine* and a *Justification Engine*, the latter assessed additional information to provide a global justification as distinct from the hitherto local justification which merely restated the collection of rules. The *Justification Engine* more accurately mimicked the human experts who may justify a decision by outlining the various rules, but is more likely to explain how the facts in a given situation supported the conclusion.

Knowledge engineers and client users both require systems to explain output, and thus the needs of the user will affect the type of explanation required. The users are more commonly provided with, if anything, a trace of rules applied; when in fact the user may be seeking a justification for the overall problem solving approach. In an attempt to tackle this problem, Ellis (1989, 123) reported on XPLAIN, in which it was proposed that domain knowledge would be represented in a descriptive domain model and a prescriptive set of domain principles so that justification of program behaviour could be provided in addition to justification of program outcomes.

Mays (1988, 559) cited several studies which have shown "questioner's expectations of the computer system are increased when the mode of interaction is natural language ... [and] may even exceed those of similar human exchanges." Gaines (1988, 271) described the changes in computing technology and user-interface over the last four decades. These are summarised in figure 2.4.14. The supplementation or replacement of keyboard and textual interaction with graphic displays and pointers have been considered beneficial. But these changes have also created problems because of their speed and frequent lack of foundation. This dilemma is compounded by the increasing variety of users, ranging from computer novices to those experienced in different systems; ranging from older people experienced in the real world but computer illiterate, to computer literate young people who lack experience in the real world. The dilemma is whether to halt the introduction of new interface technologies until the problems are resolved and the world catches up, or to introduce further advances in order to bypass the current problems and help the world catch up.

Figure 2.4.14
Dialogue style changes (Gaines 1988, 274)

Computer generation	Graphic Representing world	Formal Representing computer	Natural Language Representing person
2 1956-63	**Expensive** Flight simulators Process simulators for limited military and industrial use	**Simplistic** Job control languages giving operators access to machine features	**Output only** From stored script in early computer aided instruction
3 1964-71	**Practical** Mimic diagrams Light pens Touch screens	**Standard** Simple prompt response Menus Form-filling	**Primitive** Keyword recognition Incorporation of words from input in output
4 1972-79	**Creating reality** Windows Icons Desktop simulation	**Sophisticated** Dialog engineered prompt response Interactive form-filling Intelligent form-filling	**Practical** Understanding fixed domain Meta-level understanding
5 1980-87	**Low cost** PC flight simulators Integration through Macintosh Integration through Symphony & Framework ÷	**Integrative** Integration through INTELLECT & GURU ÷ ÷ Fifth generation objectives	**Sophisticated** ÷ ÷

2.4.4 COMPUTER SOFTWARE

Developers of an expert system may use a programming language such as LISP (John McCarthy 1959) or PROLOG (Alain Colmerauer 1972) to prepare their own programs. These languages were developed to enable mapping of natural concepts and to provide rich representation constructs to enable processing of symbolic information such as words and phrases. For example, standard PROLOG uses backward chaining and thus is especially useful for diagnostic systems. LISP and PROLOG can also provide significant programming productivity compared with earlier languages. For example, a "program that is 1000 lines long in FORTRAN can typically be written in less than 100 PROLOG statements" (Citrenbaum *et al.* 1987, 53).

Developers may also consider using a shell to provide the main architectural features of the desired expert system to which the local knowledge is then entered. The early shells were a by-product of existing expert systems in which the specific knowledge base had been removed but other structures left intact, but many shells are now purpose built (Alty 1989, 192). Shells are designed for general domains and a particular shell is unlikely to be suited to all tasks, thus requiring careful assessment of potential shells for use in a specific domain. Shells may also contain components additional to the specific requirements or, more importantly, may not have all the components necessary to satisfy the specific requirements. Newcomers to expert system programming usually start with shells on personal computers (Bielawski and Lewand 1988, 5, Martin and Law 1988, 582, Alty 1989, 199, Whittaker *et al.* 1989, 18).

A shell does, however, provide a compromise situation in which users sacrifice flexibility and complexity. Some of these disadvantages are diminished when using the more powerful shells, which are usually associated with mainframe computers. Most shells do, however, usually have important support facilities such as editing, debugging and in-built command structures. Further, users do not need formal training in a computer language to commence and maintain simple shells. Shells, like many other recent computer applications, such as word processors, do not require an understanding of how they work to use them effectively. The relevance of these factors will depend on the likely frequency with which an individual or organisation is likely to develop expert systems. Bratko (1989, 85) reasoned that the use of each new shell requires time to learn and implement whereas PROLOG already has a sophisticated formal base, its syntax is clear and semantics easily understood. However, Giarratano and Riley (1989, 17) argued against re-inventing the wheel and that it is "more efficient to use specialised tools designed for expert system building than general purpose tools". Carrico *et al.* (1989, 10) also described the benefits of using a product which presumably would have vendor support and thus be less dependent on individuals, have standard features enabling a team approach, and have relative portability.

Because shells will potentially be used by people with limited computing skills, it may be difficult for them to find the correct shell for a particular situation. To overcome this dilemma, an increasing number of articles is being published to assist developers in this decision (for example Raeth, 1990 contains six chapters of such evaluation including a comparison of twenty seven shells), and some expert systems have been developed specifically to compare shells.

A variety of Tools (also described as Toolkits) have been developed to assist in the construction of expert systems. An important group of toolkits lies somewhere between a language and a shell, to the extent that some are described as shells rather than pseudo-languages. Their use may, however, be limited by their complexity and the need for intensive training and special hardware (Alty 1989, 190 and Rothenberg 1989, 205). Williams (1986, 70) noted that "those acquiring overly simple expert systems development tools may find their caution counterproductive" because such tools usually cannot deliver the desired outcomes and thus require upgrading and retraining. Program maintenance has also been recognised as a potential problem when using relatively unknown toolkits; for this reason Matthews (1987, 433) advocated translating the finished product into mainstream languages, such as LISP and PROLOG, which could be more readily maintained by a wider pool of programmers.

The situation regarding development tools appears to have changed dramatically in the relatively short time since the above comments were made. The first large development tool was released in 1984. By 1989 approximately two dozen software development tools had appeared on the market with varying degrees of efficiency and acclaim. In 1988, the price for personal computer Tools ranged from US $100 to $10 000, mini-computer and specialised work station tools ranged from US $15 000 to $75 000, and mainframe computer tools from US $25 000 to $250 000 (Carrico *et al.* 1989, 144). By 1988 the next generation of software development tools was emerging and could be expected to improve on earlier tools. Gevarter (1987, 51) predicted that the "current tools are only forerunners of [those] yet to come" and that the future tools will be available for general computers and may be embedded into larger systems.

Edwards (1991, 72) reported a 1988 British survey which indicated that 11% of expert systems used conventional languages (such as FORTRAN), 23% used special purpose languages (such as LISP), 11% used toolkits, and 56% used shells. These statistics help to demonstrate that there is no single or simple solution in the selection of an appropriate software base for an expert system. Figure 2.4.15 summarises the main comparative features of the three approaches outlined in this thesis. Shells are intended to enable non-programmers to benefit from efforts that have already been made to solve similar problems, but shells are not suited to all tasks. Languages give experienced programmers more flexibility at low cost, with some languages including improved user interfaces and routines to make life easier for the developer and user. Toolkits are emerging to combine modular components (as in shells) and system philosophy with the kind of control normally associated with languages.

Figure 2.4.15
A Comparison of Languages, Toolkits and Shells (Based on Alty 1989, 190)

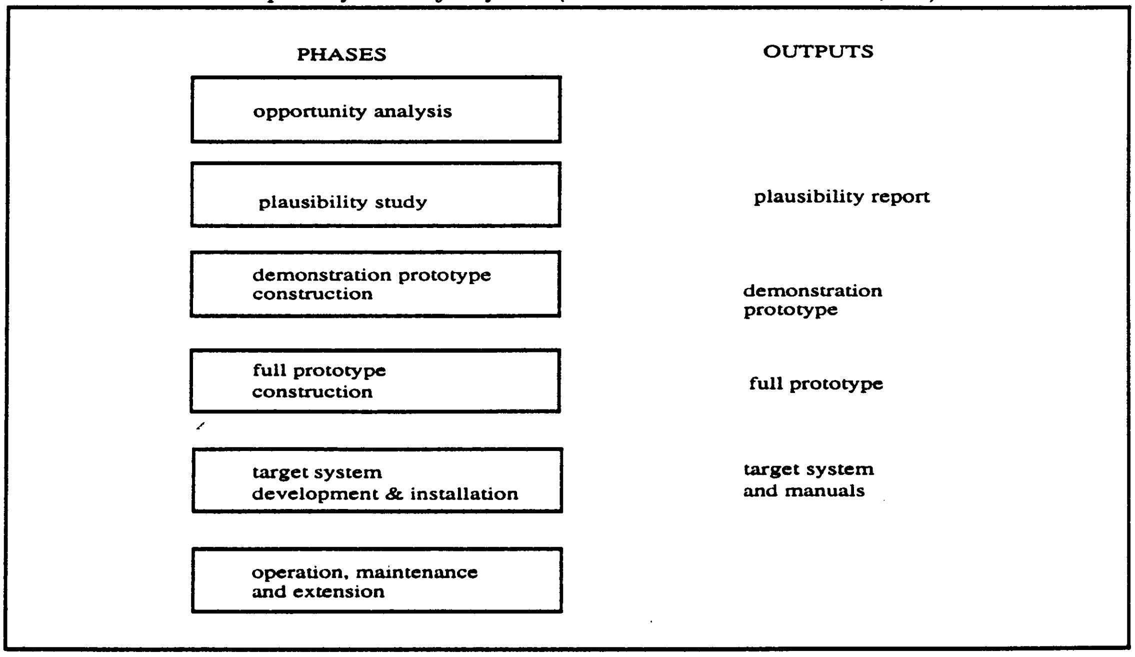
	Language	Toolkit	Shell
Applicability	wide	wide and specific	specific
Abstraction	low	high	medium
Facilities	limited	rich	medium
Hardware $	low	high	low
Software $	medium	high	medium
Training	medium	long and intensive	short

Considerable effort is necessary to ascertain which approach is best for a particular situation; that is, to match the problem with what is available. In particular, the characteristics of the domain problem, an appropriate problem-solving technique, and the desired characteristics of the resulting expert system (Waterman and Hayes-Roth 1983, 211). However it is also important to note the increasing variations and hybrids of these approaches such that the boundaries between the three approaches are often quite blurred.

2.5 INTRODUCING AN EXPERT SYSTEM

Much has been written on the problems of introducing technological change into the office environment. Problems inevitably arise when staff feel alienated by changes being thrust upon them. Successful implementation is more likely to occur when staff are involved in the planning and implementation of change, when meaningful training and counselling are provided, when respect for individual staff and their skills is maintained, and when implementation is not rushed (Bucknall 1988, 47). Many writers have suggested strategies to facilitate the effective development and introduction of expert systems through the use of rational, predetermined steps or processes. There are flow charts, waterfall models, spiral models and many other life-cycle proposals. The nomenclature and number of steps vary but most include the following stages: identification, conceptualisation, prototyping, user interfaces, testing and maintenance. Figure 2.5.1 illustrates one specific life cycle.

Figure 2.5.1
An expert system life cycle (Guida and Tasso 1989, 19)


PHASES
OUTPUTS
opportunity analysis
plausibility study
plausibility report
demonstration prototype construction
demonstration prototype
full prototype construction
full prototype
target system development & installation
target system and manuals
operation, maintenance and extension

The plausibility report would analyse a specific problem, assess the plausibility of applying an expert system and propose a draft design and project plan. The demonstration prototype is intended to provide a practical insight into the complexities which may arise and any necessary design modifications, gain participation of at least some people who would be involved in the main project, and elicit commitment to the project. These early stages are critical to the ultimate success of the whole project. There are many examples (for example, Lindsay 1988) of projects which have fallen by the way because the problem was not clearly understood, typically illustrated when people are impatient at the beginning.

The full prototype would normally be completely different from the earlier demonstrator. Whereas the demonstrator was intended to indicate potential, the prototype will contain a detailed examination of the technical components, such as the knowledge base structure and language rules. The target system should have the functional performance of the prototype but be installed in the real environment ready for routine use. The final phase represents support of the target system to ensure that it satisfies the original intentions plus developments to satisfy future requirements.

The uncertainties in conventional computer systems analysis are potentially higher in developing an expert system because these systems are essentially vague. Thus throughout the life cycle there needs to be provision for a return to earlier stages of the cycle. Such feedback loops may result in radical revisions and will at least help to facilitate fine tuning as errors and additional needs are ascertained. It is important for all the parties involved to be flexible and appreciate that these feedback loops, and any resulting changes, are healthy mechanisms and are not indicative of poor performance.

2.5.1 PARTICIPATION

Although there appear to be common elements in the stages advocated, there is less agreement on who should be involved in these developments. Surrounding the arguments that such exercises are best left to the technicians, Jagodzinski and Holmes (1989, 240) noted that the acceptability of an expert system depends on much more than the computer/user interface; it also depends on "a user-oriented, process-centred approach to the design and implementation of expert systems" which also identifies and addresses human issues.

Within the wider considerations pertaining to the introduction of an expert system into school administration, the source of the expert system is a prominent consideration: to purchase an existing product, to purchase a shell for in-house modification, to contract an external organisation to prepare a custom product, or to develop an expert system with the school's own resources. Silverman (1987, 9) reasoned that the development of the knowledge base and rules is too important to be left in the hands of engineers and was, instead, a potentially significant opportunity for direct involvement by managers to identify and correct problems within the organisation. Harmon (cited in Bielawski and Lewand 1988, 14) reasoned that managers were unlikely to get involved in creating an expert system if they had to learn all the intricacies of such a system, and thus there were considerable benefits in utilising a personal computer based shell. Nydahl (1991, 17) noted that people utilising shells on such tasks "learnt a lot in their target area whether they succeed in building a good system of not". Bonnet *et al.* (1986, 12) also presented a strong case for self sufficiency in the design, construction, operation and maintenance of expert systems. On the other hand, Babrow *et al.* (cited in Jagodzinski and Holmes 1989, 227) does not recognise a role for friendly users until the testing of prototypes. In criticising this approach Jagodzinski and Holmes cited a number of studies which reported the practical consequences of not involving end-users in the early stages. Gaschnig *et al.* (1983, 245) noted that the psychological benefits arising from the early involvement of end users "cannot be over-emphasised."

Carrico *et al.* (1989, 12) expressed the view that an organisation's decision-makers needed at least to have a background understanding of expert systems in order to make wise decisions regarding their use. Wright (1993, 48) also reflected that the people who decide which computer software will be used in organisations "generally know precious little about how the programs are used" even though the need for computer literacy at all job levels had increased in recent years. He cited an American survey which indicated that 71 percent of companies surveyed expected these skills from their managers and supervisors, yet training for such literacy remained *ad-hoc*.

Blanning (1988, 162) noted that although expert systems have in the past been developed at operating levels, "there appears to be a potential for applying expert systems in high-level staff units." He noted that the growing use of expert systems by corporate headquarters is likely to be mirrored in the US Army. A beneficial part of this development would be to ascertain what high-level staff officers and analysts actually do. McGraw and Harbison-Briggs (1989, 20) argued that, beyond research situations, it is unlikely that an individual will have the necessary skills to be domain expert, knowledge engineer and programmer, as individuals highly skilled in one of these areas is unlikely to be trained in the other areas. But Shin (1988, 304) reasoned that the answer lies in selecting appropriate solutions for appropriate problems so that amateurs can with minimal training develop practical expert systems.

"People who believe that expert systems can be built by unskilled personnel following some simple recipe are headed for disappointment" (Jackson 1990, 366). Jackson expressed this view after examining a number of surveys which confirmed his own experiences with post-graduate students. Indications are that learning to use complex shells and toolkits is no easier than learning a new programming language. Since programming is not everybody's forte, it is inappropriate to suggest that anybody can develop an effective expert system; but this is not to suggest that inexperienced people should be discouraged from considering such involvement. Indeed, the continued development of shells and toolkits will undoubtedly make it easier for people working in a given field to become knowledge engineers. Nydahl (1991, 16) noted that knowledge engineers become "some sort of expert within the area" and that, with the development of better expert systems, it was being demonstrated that experts could be successful knowledge engineers. Edwards (1991, 4) stated that people do not have to be "steeped in AI ... [to] make use of the fruits of AI research", any more than people using conventional programming need to undergo theoretical research in mathematics. Edwards also noted (1991, 83) that people with a background in conventional computing sometimes found the non-procedural aspects of expert systems difficult to accommodate.

Finkelstein (1988) approached this debate from a different direction, describing a family of Australian expert systems designed to enable managers without a knowledge of computing to generate Expert Business Systems that automatically designed and generated knowledge bases. These systems which had been used by several businesses and government groups, were described as a revolution in the way computer applications are designed and implemented - especially because the amateurs have to be involved.

Whoever is involved, but especially amateurs, should not ignore the underlying truth that effectively programming an expert system requires the same discipline as programming other computer systems and therefore they need to address matters such as "data modelling, quality assurance, quality control, testing, modular design, maintenance, etc." (Carrico *et al.* 1989, 14).

2.5.2 FEASIBILITY

Two major aspects need to be examined when considering the introduction of an expert system: the problem to be tackled, and the resources available. Not only does the problem need to be compatible with expert system software and hardware, it also needs to be worth solving with an expert system. Problems should be neither too simple (less than ten rules) nor too complex (more than 10,000 rules) and are most appropriate if the domain is narrow but deep. An expert system should be considered where it is necessary to save time, preserve endangered knowledge, be used as a training device or be used in a hostile environment. Expert systems may also be considered for situations where human experts would not have time to consider a vast and rapidly changing array of variables. For example, 500 warning lights were activated in the first minute of the Three Mile island nuclear power station incident (Bauer 1983 cited in Paterson and Sachs 1989, 218).

Edwards (1991, 33) warned against solution-driven decisions which can arise, especially when, as the result of increased publicity or commercial endeavours, people presume that the new, exciting and apparently successful products will provide the solution to as yet undefined problems.

It is generally recognised that expert systems should not be used in situations which could be effectively solved by conventional programming; that is, where there is an efficient algorithmic solution. Nor should expert systems be considered where the benefits do not warrant the hardware, software and other costs. Giarratano and Riley (1989, 21) suggested that while tackling apparently ill-structured problems, knowledge engineers may "unknowingly discover an algorithmic solution", in which case the problem "may be a good candidate for recoding as a conventional program". But why? If the expert system is at least efficient as a conventional program but retains the option to include ill-structured problems in the future, then it may be a good candidate for leaving as an expert system. Reynolds and Cartwright (1989, 155) noted that expert systems may yield improved performance over conventional systems even when an algorithmic approach is possible. They supported the notion of exploiting a wider system and cited a successful project for a ship's propulsion system involving communication with human operators as well as data sources and control interfaces. They also reasoned that a "well designed knowledge based program should be easier to maintain and easier to re-use in new circumstances than a conventional program."

Three main resource issues need to be confronted prior to commencing such an exercise. (1) Personnel: is there a source of knowledge — an expert, is there a knowledge engineer, can someone implement the expert system? (2) Hardware: what is or can be available, what is required by the shell or language, are development and real time environments compatible? (3) Time: can all involved commit sufficient time, can someone maintain the expert system?

The selection of a domain is a task critical to the ultimate outcome of a project and the effort which needs to be devoted to this selection should not be lightly dismissed if one is to ensure that potential problems are minimised or eliminated (Prerau 1989, 27). This selection process includes not only the evaluation of the domain to which the expert system is to be applied, but also includes an indication of the desirable attributes of the intended expert system. Potential users of an expert system would do well to apply their situation to the pseudo expert system in figure 2.5.2 to help ascertain the appropriateness of the domain and the use of an expert system in this domain.

Figure 2.5.2
A rule set describing when to use an expert system (Based on Silverman 1987, 10)

```
Goal : Determine if an Expert System approach is appropriate
        IF      the ES approach is
                        relevant, AND
                        feasible, AND
                        optimal, AND
                        success oriented
        THEN an ES approach will be appropriate (cf 100)

        IF      there is a recurring shortage of skilled employees, OR
                problems regularly require innumerable solutions, OR
                job excellence requires unreasonably high levels of training, OR
                no single person can know all the expertise, OR
                management keep applying existing knowledge to
                        basic problems,
        THEN ES is relevant (cf 85)

        IF      the problem typically takes time to solve, AND
                no controversy over problem domain rules exist, AND
                problem domain experts exist, AND
                knowledge can be expressed,
        THEN ES appears feasible (cf 85)

        IF      it is necessary to make heuristic judgements, AND
                ES software is more appropriate than conventional, AND
                human interaction is required, AND
                reasons must be given for asking questions, AND
                the answer must be justified,
        THEN an ES is the optimal approach (cf 100)

        IF      solutions are of high value, AND
                management supports an ES approach, AND
                a prototype can be constructed, AND
                it can be incrementally implemented, AND
                the knowledge engineering team have a successful record,
        THEN ES approach is likely to be successful (cf 75)
```

If the four conditions are satisfied, by their own second level rules, then the goal can be satisfied and an expert system approach would be appropriate.

The domain selected, 'student subject selection', is very important to the end users, the students, for their future study, for their personal ambitions, and for their careers. Ainley *et al.* (1994) reviewed studies, undertaken since the early years of the twentieth century, of school subject preferences and subject choice and noted that

> there is now a great deal of psychological and sociological information ... [which] suggests that interests are very strongly related to subject preferences. These interests seems to activate students in their choice of both school subjects and their preferred occupations. Clearly, local factors - timetabling, school size and so on - can limit a student's choice (19).

Although students in the Northern Territory need to comply with defined subject time allocations to attain their Junior Secondary Studies Certificate, there still remains sufficient flexibility for students in some schools, including this school, for subject selection decision making. Subject selection in junior secondary schooling is important, not the least because senior secondary patterns "do not simply arise from decisions taken at Year 11 and Year 12 but as a result of gradual focussing over time" (Ainley *et al.* 1994, 3). Davies and Ellison (1992, 13) also noted that those market forces which have accompanied the devolution of responsibility in schools have started to move the attitude of schools from being "product-orientated to being more client-orientated", thus ensuring that schools adopt a high-quality, service approach for their products so as not to let clients down. Schools which offer a unitised curriculum through a vertical timetable do not provide the open-learning, personalised schedule favoured by English (1993) but are more responsive to individual student needs than schools which adopt the timetabling strategies of Lewis (1961, 4) who noted that

> the majority of boys present no difficulties as the courses they choose are of a conventional kind; it is only the occasional eccentric wishing to combine ... subjects from different sides who is liable to be awkward.

The organisation of subject selection has also become more complex through increased student numbers. Retention rates to Year Twelve in Australia rose over the decade to 1989, with the most dramatic increase in the Northern Territory where they nearly doubled from 22.2 to 42.7 percent (Department of Employment, Education and Training 1989). 1993 projections indicate some minor fluctuations at various year levels but predict a general trend for increased enrolments in the Northern Territory over the next decade (Department of Employment, Education and Training 1993).

Subject selection is of central importance to the students and thus must be of central importance to the organisation of the school. Curriculum content, student assessment and accreditation, subject availability, increasing enrolments and higher community expectations of service providers are all factors which increase the pressure on schools to provide an effectively managed education for students. The domain is therefore an acknowledged problem area for which an expert system may provide significant benefits.

2.5.3 Evaluation

Evaluation should be a component of the overall implementation program and not simply responses to problems that may arise. Formal and informal evaluations should "pervade the system building process [as they] are crucial for improving system design and performance" (Gaschnig *et al.* 1983, 242). To help ensure that a project remains on task, and to facilitate confidence in the end product, evaluation requires feedback from at least the domain experts and end users to ensure that what goes in and comes out complies with the requirements of these two groups.

Gupta (1991, 200) noted that faults in expert systems often arise from two factors: "the developer's reluctance to invest project resources in writing specifications, and the user's ignorance regarding what an expert system can and should do". Bench-Capon *et. al.* (1993, 76) noted that the construction of knowledge bases is typically "too shallow" and may be biased by failing to recognise the actual boundaries of the expertise embodied in the domain. Other authors, such as Green (1988), have described difficulties that inevitably arise when the requirements of an expert system are not documented and monitored, and have advocated the appointment of a Requirement Specialist to be responsible for this co-ordinating role. Documentation often appears difficult, especially when the goal of the intended expert system is not clearly understood by the customer (for example, "we want a program to do what Sam does").

As in conventional programming, there are significant benefits in planning and reducing unnecessary errors and misdirection. An early evaluation should be undertaken to confirm the project's intended outcomes and determine the feasibility of developing an expert system to satisfy these aims. If the project proceeds then its implementation model should be evaluated. Finally, the expert system needs to be evaluated during construction and on completion, with on-going monitoring.

Implementation models should include provision for evaluating the expert system before it becomes a critical function within the organisation. Evaluation may include the use of a prototype to facilitate clarification of a system's characteristics and operations through the construction of a scaled-down working version. This is especially important when the user's requirements are not easily understood and thus subject to change as the system is developed and demonstrated.

Prototypes may be paper or computer models of the intended expert system, but necessarily in an incomplete stage of development commencing with a minimal but functional system. Because some components have been simplified, the prototype will be unlikely to truly reflect the finished product. Many authors, such as Keller (1987, 111) have outlined the concept of a rapid prototype in which models are incrementally developed. Not only do rapid prototypes reflect typical human development of establishing then refining procedures, but also avoid a long gestation period before something can be demonstrated and evaluated. Rapid prototypes result in early models and concepts and thus become a basis for further thought and refinement. Olson and Lindahl (1988, 84) cited Texas Instruments (1980) statistics that more than 70% of errors are not detected until a computer system is fielded, even though more than half the errors occurred during the design stage. Rapid prototyping helps to identify errors more quickly and reduce the cost of correcting them. Kahn and Bauer (1989, 59) reported informal indications that the use and impact of rapid prototypes was encouraging and expected to increase.

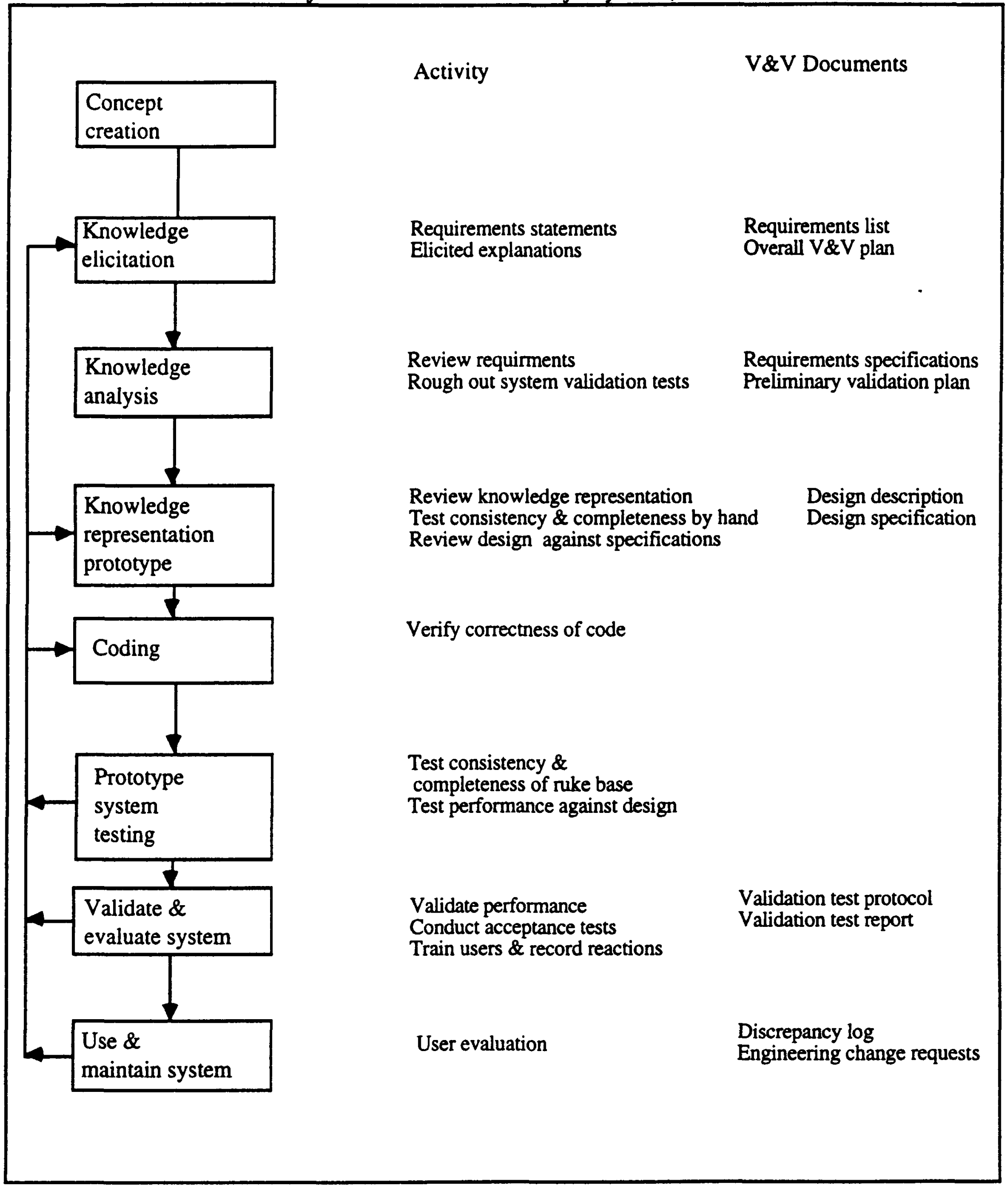
Hollnagel (1989b, 411) noted that the evaluation process has a tendency to focus attention on situations where the expert system outcomes differ from expectations and tend to ignore situations where the outcome is as expected, without evaluating these results. Hollnagel explained a clear distinction between evaluation methodology and system evaluation, and concluded that current expert system evaluation techniques are too few and *ad hoc* to provide an adequate methodology in this field. To avoid the pitfalls of evaluation being misunderstood, it is important that the purpose and expected benefits of formal evaluations are understood by all concerned. Inherent in this is the need to ensure that the parties involved know what is being evaluated, why and for whom.

Many authors distinguish between verification and validation. Gupta (1991, 1) described verification as determining that the system was built right, whereas validation determines that the right system was built. Verification to ensure that the actual coding is free of errors, a process greatly assisted by computer programs which have in-built debugging facilities, and validation to ensure the adequacy of the program output to meet the intended objectives and standards. Some authors, such as Hollnagel (1989b) and Guida and Mauri (1993), also distinguish between evaluating the construction of an expert system and assessing whether the expert system is actually used and any ramifications of its use in the context where it is applied.

Evaluating expert systems will often be more complex than evaluating conventional programs because of the inclusion of heuristics and the lack of precise algorithms. On the other hand, Culbert *et al.* (1987, 233) commented that the separation of knowledge and procedures may actually make testing the knowledge base easier. Gupta (1991, Section Two) described a number of attempts which have been made to develop systems for knowledge base verification. These systems were designed *inter alia* to evaluate interaction, syntax and semantics. Some systems can take remedial action to correct the knowledge base, others produced a variety of reports. In most cases, however, the expert system had to be at least functional, and in some instances in use, before the debugging system could be used.

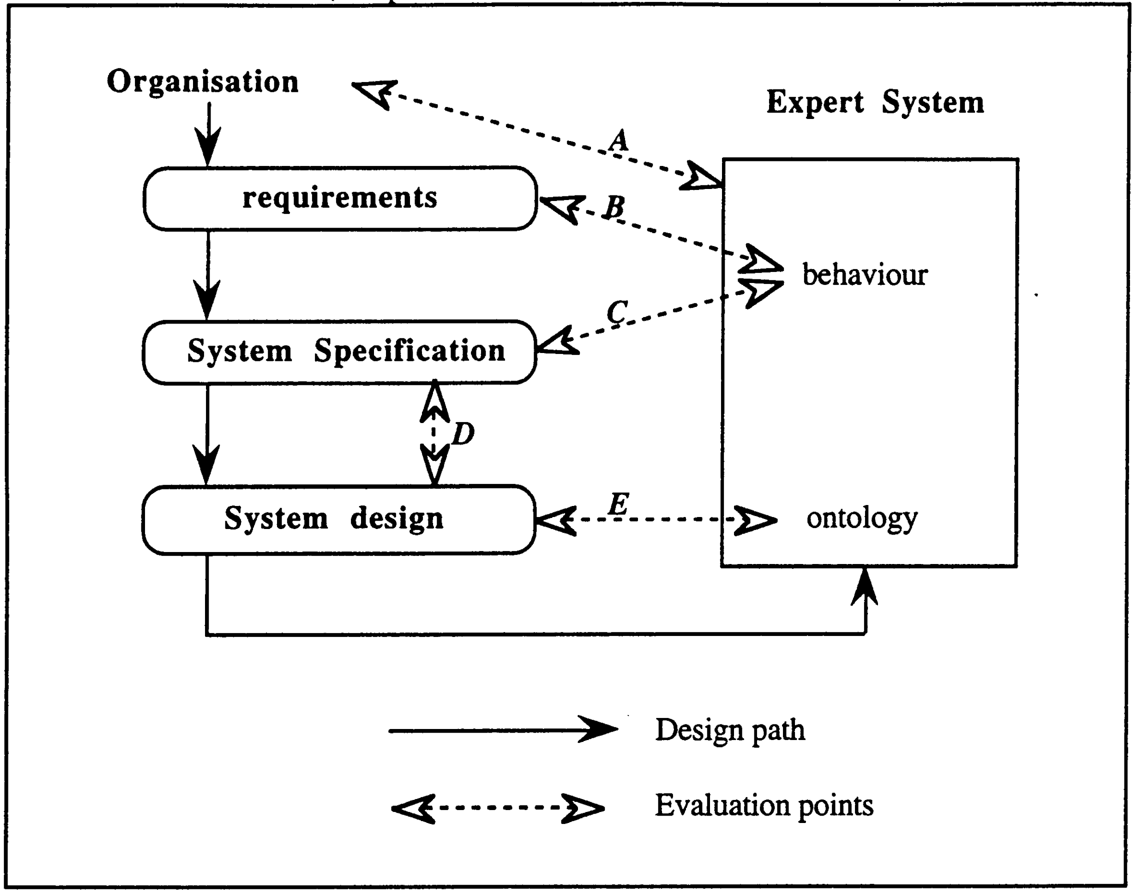
Liebowitz (1986, 249) reflected that "there has been a myriad of approaches, through the years, on how to perform evaluation". O'Leary (1987, 57) summarised the role of validation as "1. ascertaining what the system knows, does not know, or knows incorrectly; 2. ascertaining the level of expertise of the system; 3. determining if the system is based on a theory for decision making in the particular domain; 4. determining the reliability of the system". Bielawski and Lewand (1988, 272) identified six specific criteria for testing and validating an expert system: accuracy, completeness, reliability and consistency, effective reasoning, user friendliness, and run time efficiency. O'Keefe *et al.* (1987, 3) examined seven major problems encountered in evaluating expert systems: (1) what to validate, (2) what to validate against, (3) what to validate with, (4) when to validate, (5) how to control the costs of validation, (6) how to control bias, and (7) how to cope with multiple results. O'Leary (1987, 59) noted that validation should not be restricted to the actual expert system but needs to include evaluation of the user interface, documentation, as well as the language - shell - toolkit used to develop the expert system. While Rushby (1988, 79) distinguished between the desired and minimum competency requirements for expert system performance as distinct from the service requirements for the expert system development and construction. Naser (1988, 30) described several evaluation life cycles (such as figure 2.5.3) to accommodate the different (from conventional programming and from each other) needs of types of expert systems to verify their knowledge base for consistency, completeness and correctness.

Figure 2.5.3
A verification & validation life cycle (Naser 1988, 40)



More recently, Guida and Mauri (1993, 204) reviewed the work previously undertaken in this area and proposed what they described as "a novel approach to [expert system] evaluation which comprises a foundation of the concept of evaluation and a general evaluation methodology." This approach is summarised in figure 2.5.4.

Figure 2.5.4
Expert system design and evaluation
(Simplified from Guida and Mauri 1993, 210)



Guida and Mauri reasoned that there are five evaluations which need to be undertaken to ensure performance and quality of the system. These five evaluation points are shown in figure 2.5.4: (A) Assessing the actual system use and utility within the organisation, (B) Validation of the systems requirements and actual behaviour, (C) Verification of the systems specifications and actual behaviour, (D) Evaluation of the system design against its specifications, and (E) evaluation of the constructed systems ontology (foundation principles) against the design intentions.

Just as it is sometimes difficult to evaluate an expert's performance, it may also be difficult objectively to evaluate an expert system. This problem is compounded when determining acceptable standards of performance. A 100% performance by an expert system is probably unrealistic if the human experts perform at a significantly lower level. Chandrasekaran (1983, 261) noted that the success-failure dichotomy is insufficient in that it ignores intermediate performance which may be very acceptable. Evaluation needs to consider not only how well an expert system works but also how badly it can fail before such failure presents a meaningful problem. Rushby (1988, 77) reasoned that although errors should be avoided, end-users will inevitably be faced with a range of fault tolerances from benign to catastrophic. The significance of fault tolerance will vary between tasks; for example, a benign fault in air traffic control may have catastrophic consequences while a catastrophic fault in an expert system for selecting fishing lures may have relatively benign consequences. In some cases of student subject selection there may not necessarily be a correct answer, though some answers would be clearly incorrect.

Blanning (1987, 28) and Whittaker *et al.* (1989, 19) recommended that evaluation should include both open book and a blind validation (also described as the Turing Test); the first by people who knew they were comparing human and expert system recommendations and the second by a person who did not know which was which but was asked to comment on a mix of human and expert system recommendations. However, several studies have indicated a potential problem with open book evaluations: viz, bias against computers by those judging the recommendations (Gaschnig *et al.* 1983, 263). Other studies cited by Gaschnig *et al.* (1983, 249) have indicated that comparing the recommendations made by an expert with those from an expert system ignore a bias which may work in favour of the expert system dealing with a specific domain and not facing broader distractions. On the other hand, the restricted domain of the expert system may reduce its performance in situations where a richer or broader understanding is required, though perhaps this reflects a poor choice of domain for the expert system, or its inadequate preparation. Kulikowski (1989, 174) also reasoned that the expert system's performance is only "a small part of the overall evaluation", which also needs to take account of its integration into current arrangements and its economic and social impact.

"Perhaps the ultimate [evaluation] is whether an expert system is actually used for expert consultation by individuals other than the system developers" (Gaschnig *et al.* 1983, 245).

2.5.4 Reliability

Related to, yet distinct from, validation and verification is the problem of reliability. Bundy (1989, 43) listed four potential aspects of unreliability: (1) fragility evidenced by the expert system failing in unexpected ways, (2) upredictability of the answers, (3) brittleness and non-flexibility in new problems, and (4) discontinuity evidenced by significantly different outputs from similar inputs. He observed that some of this unreliability stems from poor practices when developing expert systems. For example (1) mixing controls and facts in the same rule, (2) attaching arbitrary procedures to rules, (3) multiple knowledge representation without appreciating relationships between these, (4) uncertain use of certainty factors, (5) incremental development without adequate consideration of the overall expert system, and (6) a lack of theoretical understanding of the system and its development.

2.5.5 MAINTENANCE

A key issue in developing an expert system will be provision for updating its knowledge base. Software maintenance modifies an expert system without changing the primary functions of that system. Despite effective validation and verification, it is normal for expert systems to require ongoing maintenance, if only because expert knowledge rarely remains static. Some predictions are that the cost will be more than half the project budget (for example, 60-70% Carrico *et al.* 1989, 219). Some of this ongoing maintenance may be to correct errors in or to enhance earlier versions; but it should be reasonably expected to incorporate new knowledge or expertise since earlier elicitation. Edwards (1991, 99) stated that maintenance "should be the longest of all the phases in the life-cycle as far as elapsed time is concerned".

At some point the distinction needs to be made between maintenance and functional updates. For example, XCON an expert system for Digital Equipment Corporation grew from 700 to over 6200 rules over seven year's use (Carrico *et al.* 1989, 220), with the danger of such changes resulting in incoherent, dead or lost code. Thus even if maintenance is undertaken by the system's original developers, and especially because it is likely to be undertaken by other people, it is critical for expert systems to be structured and well documented.

It will also need to be determined, preferably beforehand, who is going to control the expert system and take responsibility for its contents and the impact any changes may have on recommendations. There are several approaches to maintenance, including: a) keeping it separate and independent from the current system, despite the need to have more people who understand the system; b) using domain experts if the system is relatively stable and easy to modify; and c) providing for and using end-user responses (Carrico *et al.* 1989, 222).

2.6 GENERAL CONCERNS

Characteristic reactions by people, when first acquainted with the concept of expert systems, fall into one of three genre: impressions that here is another science fiction scenario coming to realisation; positive and negative personal recollections of other computer experiences; and vague ideas on how an expert system might effect them. Initial reactions are then followed by in-depth considerations which lead to a number of concerns: (1) whether a computer can and should replace people; (2) whether the advice can be trusted, (3) how it should be used and by whom; and (4) the impact on employee esteem and employment.

2.6.1 WHETHER A COMPUTER CAN AND SHOULD REPLACE PEOPLE

Pohl (1984) presented a taxonomic hierarchy (summarised in figure 2.6.1) analysing the effect of machines on our social fabric and with which she examined, *inter alia*, the necessary shift in people's view of self and their interaction with machines that have human characteristics. She concluded that there are a number of possible outcomes which could be beneficial or threatening. Hext (1991, 15) also observed that the visible and immediate effects of new technologies "are ultimately far less important that its subtle side effects" and thus an examination of new technologies needs to examine the long-term impact on society. He reasoned that it is necessary to strike a balance between the technological mindset and Luddite philosophies to find an appropriate middle ground. Bucknall (1991, 13) reflected that "the future will not be a projection of the past and if we are to empower schools then they must be proactive to cope with change".

Figure 2.6.1
Impact taxonomy (Pohl 1984, 290)

Type	**Examples**	**Consequences**
Methodological	Word processors Automated tellers	Improved productivity but essential skills not significantly impacted
Dislocating	Industrial robots Electronic functions	Large scale shifts in jobs and life styles
Paradigmatic	Artificial intelligence Socialized machines	Redefinition of Man's role or essence

The process of translating someone's expertise onto a computer knowledge base will inevitably express that knowledge as a phlegmatic collection of facts and opinions devoid of the contextual and informal feedback that the human expert experiences. It can be reasonably argued that much of the expert information is lost in this process (Australian Science and Technology Council 1987, 9). Some writers argue that if a computer system simulates intelligent behaviour then it provides a model of human intelligence, but Wells Coleman (1987, 781) strongly advocated the need to emphasise the artificial component of artificial intelligence. He expressed the view that "just because a computer looks like it is doing the same thing as a human ... does not mean that it is". To reduce this problem, some authors (especially in the United Kingdom) advocate the nomenclature Knowledge Base Systems to avoid the expert connotation. Weizenbaum (1976, 203-223) stressed that "computers and men are not species of the same genus" and that it is not a question of technical feasibility but, rather, a more fundamental issue as to whether "it is appropriate to delegate this hitherto human function to a machine" since "no other organism, and certainly no computer, can be made to confront genuine human problems in human terms".

Decision making typically involves more than an impersonal expertise and human experts are normally held responsible for their advice. Jagodzinski and Holmes (1989, 229) cited Hollnagel who identified problems which could arise if experts reduced their personal interaction and relied on expert systems as their main decision support system, especially if these experts were unable or unwilling to challenge the system's advice. Michie (1982, 253) also highlighted the potential danger of an increasing dependence on computer systems which rapidly evolve beyond the understanding and control of people.

However, the significance of these drawbacks will depend on one's definition of artificial intelligence in general, and expert systems in particular. If the computer software is intended to replicate the human expert, then the drawbacks are significant. But if the computer software is only intended to mimic the advice provided by that human expert, that is, is not intended to provide a model of human intelligence, then providing the advice is correct, the drawbacks are insignificant. Developers have not devised programs that will enable computers to do tasks that no human knows how to do; but they have devised programs that will enable computers to do some tasks better than humans and in this sense an expert system could excel over its creators (Silverman 1987, 8). Silverman also noted that people can only apply their own expertise to a situation, whereas an expert system is able to apply several people's expertise simultaneously. Little wonder that there is a demand in some areas for computers to replace humans. But there is concern that we are already highly dependent on machines and are increasingly dependent on computers, and therefore should question the wisdom of further increasing our dependence on machines (Keir 1987, 15). The Australian Science and Technology Council (1987, 7) recommended that expert systems have in-built features that make it clear to users that the machine is only a tool to assist them. Michie (1982, 135) aptly described computer systems as another example of *idiot savants* which are able to perform miracles in a specific domain yet are otherwise subnormal - powerful but not intelligent. Light (1992, 134), less kindly perhaps, described the computer as "that brilliant fool of modern times".

If the role of an expert system is clearly understood as assisting decision makers determine the best course of action, then such systems will concentrate on formulating the problem clearly to gain insight into the decision problem rather than producing the correct decision (Holtzman 1989, 7). Lovie and Lovie contend that people should not compete with computers, which hold and process large amounts of information; rather, experts are "better employed in adding meaning and significance" (1989, 92).

2.6.2 WHETHER THE ADVICE CAN BE TRUSTED

Without a doubt, the advice provided by an expert system could be dangerous, have side-effects, traps and false assumptions, in exactly the same way that any human expert advice could be dangerous, especially if taken out of its proper domain and applied in another. Providing the computer expert system has been carefully constructed, rigorously evaluated, and presented as a mechanical tool then the danger should lie not in the tool but the manner in which it is used. However, even thorough testing cannot guarantee reliability as such testing will only reveal bugs, not their absence (Denning 1986, 422). This situation is compounded in expert systems, which are based on opinion as much as fact. However, there is a number of strategies to reduce at least the potential misuse of expert systems. One approach is to use the term knowledge systems rather than expert systems. A knowledge systems nomenclature is less likely to give the impression that the computer system learns from its environment in the same way the human experts are continually learning and updating their expertise.

People nowadays are aware of and use computers as a relatively common occurrence. They are used to systems which give a definitive answer according to the laws of mathematics and there is a danger that they will trust their expert systems to give an answer that is true. When human experts are confronted with a problem outside their domain, they can often apply some of their expertise and arrive at a reasonable solution, a solution for which their common sense will provide some indication of apparent validity. On the other hand, a computer expert system does not know when it might be wrong because it is unable to transfer the general principles from one domain to another and its lack of common sense will prevent it from graceful degradation when confronted with a problem outside its domain and thus lacks the human robustness when dealing with problems (Barr and Feigenbaum 1981a, 10, Amarel 1984, 3, Hollnagel 1989a, 22, Whittaker *et al.* 1989, 18).

The integrity of the database these systems use is critical in determining the quality of the advice the system gives out and thus it is necessary to ensure that the expert system's knowledge base is explicitly open and exposed to scrutiny. Expert systems will frequently depend on knowledge which is changing and thus it is important that provision be made for this knowledge to be updated over time and by people other than the system creators. Bachant (1988) described the RIME project, which was developed as a methodology to keep track of changes to the knowledge base and their compatibility with stability of the system. Some systems, such as EMYCIN, automatically record the author and date against any rules that are added or modified (Buchanan 1983, 150). Great care must be taken in developing these data bases and the very explicit rules needed to treat the information for, unlike human experts, machines do not have any tacit or background knowledge which can guide the interpretations (Hollnagel 1989a, 22). Unlike humans, expert systems do not resort to reasoning from principles, analogies, or common sense (Silverman 1987, 19). Keir (1987, 15) also noted the concern that proprietary expert systems may incorporate a bias towards certain products or outcomes, for example, encouraging boys to do woodwork and girls home economics.

An expert system which has a high internal reliability should reasonably provide users with a reliable output. But Silverman (1987, 19) and Hollnagel (1989a, 15) noted that "high reliability of the individual parts of the expert system does not necessarily produce a good result, because the way they interact may be defective". They consider that the combined input may lead to conflicting output and that an expert system may develop its own tunnel vision.

Though most expert system's commentators insist that the ultimate responsibility must be the user's, there is still a responsibility by the developers to ensure that the systems work as they should (Australian Science and Technology Council 1987, 9, Keir 1987, 14). The responsibilities of the experts, knowledge engineers, programmers, manufacturers, retailers and users may yet be determined by the courts. Thus far, expert systems do not appear to have been involved in legal cases involving negligence or product liability. But given the general trend in many countries to increased litigation, it is probably only a matter of time before a legal ruling will be sought. Schwartz (1991, 5) examined some of the problems in determining ownership (copyright and patents) and liability. He cited the example of using a licensed system applying scientific principles to design a new automotive air-bag and raised questions about who owned the design and was responsible if the product was incorrectly designed. Zeide and Liebowitz noted (1987, 452) that tracing liability back to a human source will prove difficult given the number of people usually involved in producing an expert system, and thus "the law will probably hold manufacturers responsible". Litigation vis-a-vis educational administration has been relatively scarce in Australia thus far and the use of expert systems has the potential to complicate future claims, when trying to establish responsibility for advice. On the other hand, the introduction of expert systems has the potential to reduce the incidence of poor decisions thereby reducing future claims.

2.6.3 HOW IT SHOULD BE USED AND BY WHOM

Expert systems have usually been developed to undertake at least one of the following tasks: 1) provide assistance as part of a wider task; 2) critique a decision, especially looking for possible inconsistencies or omissions; 3) working parallel to a task and providing a second opinion; 4) offering advice during an interactive exchange; 5) training staff in a given domain; and 6) independently monitoring a situation, ranging from mechanical equipment through to stock exchange trends (Edwards 1991, 16).

The most prevalent rationale for expert systems is they should act as a decision support system for people and therefore the best balance between people and the computer expert system is when the computer is used in those narrow activities where it is more skilful to support the intelligent skills undertaken by the human. Expert systems should be used as another tool (Australian Science and Technology Council 1987, 6, Bielawski and Lewand 1988, 8). Thus, rather than replacing people, the expert system can assume the role of an intelligent assistant to make people more productive. Following the computer's analysis of the situation, it can extend the user's ability to consider alternatives, draw attention to other factors that might have a bearing on the decision to be made, and to inform the user of its conclusions and, most importantly, be able to explain its inference process and/or conclusions (Silverman 1987, 19, Bielawski and Lewand 1988, 17). Thierauf (1988, 15) supported this concept and extended it thus : “Such systems assist organisational personnel in reaching effective decisions that contain elements of subjectivity and objectivity ... the capability of combining subjectivity (individual judgement) with objectivity (the computer output) permits a more thorough exploration of the problem”. Roth and Woods (1989, 238) reported that expert systems which treated their user intelligently through active human participation, rather than retarding their input, resulted in more successful and rapid solutions. Appropriate training should make it clear to users that the program is only a tool offering advice.

Another strategy is to avoid using language which implies that computers perform like humans (Australian Science and Technology Council 1987, 7). On the other hand, Finin *et al.* (1986, 279) argued strongly in favour of a natural language interaction between users and the computer system, to the extent of not only using user-friendly language beyond syntactically sugared interchanges, explanation and reasoning, but also providing smarts to increase the programs tolerance to input errors. They reason that these critical needs are even more important as expert systems move beyond well-formed problems.

One argument concerning who should use expert systems is that only experts in the domain should have access because these are the only people with the expertise to evaluate the advice. On the other hand, these are the very people who are unlikely to need the expert system. Thus novice users could be trained and tested to reach a standard of expertise in the use of a complex expert systems before they were allowed access (Australian Science and Technology Council 1987, 8, Keir 1987, 15). However, these formal restrictions should be unnecessary as expert system developers are able to define access, if only through ease of operation. One approach may be to limit the more complex systems to those users with a higher level of expertise and the ability on the part of the user to assess the accuracy and value of the advice provided. Cuff (1982, 6) recommended the use of several query languages providing a level of interface appropriate to individual users, especially interfaces with many heuristic aids for casual users needing sympathetic help.

2.6.4 Impact on employee esteem and employment

It is their expert knowledge accumulated over many years that has enabled many people to achieve their current status. For some, at least, there has been reported (Kraft 1985, 45, Light 1992, 135) a loss of status and esteem when their relatively independent work style has been replaced by an apparently less than self-sufficient output which requires consultation with a computer and possibly a remote data-base located back at the office.

Statements which express, or even imply, that expert systems will replace human experts may create resistance by those human experts to participate in the development of such systems for fear of creating their own obsolescence. Even if the potential to displace them is not actual, the "danger is that people may actually come to accept that in some sense they are becoming obsolete" (Falk and Aungles 1987, 18), which may result in experts drifting away from that domain and perhaps not being replaced by new entrants to the workforce. This does however raise the problem of the devalued skills of employees who no longer are the expert. Keir (1987, 14) posed the question of what is fair treatment for these employees whose skills are now distributed for others to use?

Jagodzinski and Holmes (1989, 231) noted that organisational equilibrium can change significantly in both task design issues and job design issues, ranging from the nature of tasks through to control and status, with the potential to create significant rifts within an organisation. Many solutions to this problem have been established in a wide variety of user-oriented approaches in the analysis and design of systems. Put simply, it is critical for the success of any introduction of new technology, especially that as complex as expert systems, that the impact be carefully managed and not simply put to one side in favour of tackling technical problems. Brulé and Blount (1989, 169) conclude that to ignore the organisational issues, which may be central to the expert's situation, may be to endanger the project altogether.

2.7 CONCLUSION

Despite the usually optimistic lauding of expert systems, especially in magazines, there are difficulties which impede the development and implementation of expert systems as effective practical decision support systems. Some of these problems relate to technical issues of programming, some relate to the cognitive aspects of developing knowledge bases, and some problems relate to epistemology and how the human experts use meta-knowledge. However, although the nomenclature Expert System may be pretentious, and common predictions for expert systems naive, there is a rapidly expanding and rigorous endeavour to develop cost-effective expert systems to provide decision support systems.

Expert systems have developed from the research into artificial intelligence, and provide a more sophisticated decision support system than the earlier management information systems. Like many other computer applications, relatively powerful expert systems are available for personal computers. Instead of using a programming language, such as LISP, expert system developers increasingly have access to shells, which provide the main architectural features and user interface procedures, and into which local knowledge is entered. Shells offer the potential for novices to try ideas at a low cost and with little training. But unlike some other software such as word processors, standard shells have not yet effectively replaced programming in serious and/or complex domains.

Knowledge can be represented by a series of relationships which cater for degrees of certainty and imprecise human input. These relationships may be expressed as rules, decision trees, frames or hybrid combinations of these. The use of certainty factors and/or fuzzy logic are two of the important elements which usually distinguish expert systems from other computer applications. The opportunity for forward and backward chaining is particularly important in mimicking interaction with human experts when the user may wish to know the requirements to obtain a given goal or, alternatively to ascertain one's options from a given starting point. Another key element in humanising expert systems is the provision of explanations for recommendations made by the computer. These explanations should extend from the source of its knowledge, the procedure followed for utilising this knowledge, and the ability to critique suggestions by the user.

There remains opposing views regarding who should be involved in developing expert systems. Although there is general agreement that senior staff should participate in the overall design, some authors argue strongly that the users should leave the system's development to the knowledge engineers and other computing professionals. However, the increase in user-friendly expert system development tools appears to be breaking down the barriers and it is becoming more feasible for amateurs to be actively involved in developing an expert system.

Many writers have discussed strategies for the effective introduction of new technology, including expert systems. Typically these include feasibility studies, demonstration prototypes, construction, installation, evaluation and maintenance phases.

Not all decision support tasks are considered suitable for expert systems. Despite the potential benefits which expert systems offer a decision support system, there are reservations regarding their technical and moral appropriateness. Thus it is important that expert systems are able not only to provide accurate advice but also that the source and rationale of this advice be readily accessible to users. In the final analysis it is critical that expert systems be recognised as decision support systems and not be used as decision makers.