INTEGRATED KNOWLEDGE SUPPORT ENVIRONMENTS FOR RAPID PROTOTYPING OF EXPERT SYSTEMS

(Invited Paper)

Brian R. Gaines

Knowledge Science Institute
University of Calgary
Calgary, Alberta, CANADA T2N 1N4

ABSTRACT

This paper introduces the concept of knowledge support environments—integrated interactive knowledge acquisition systems and expert system shells—that can provide a specialist user community with tools supporting a wide range of knowledge processes. These system extrapolate the trend from human knowledge engineering, through automated interviewing of the expert, to continuing on-line access to both knowledge acquisition and application processes. In knowledge support systems the distinctions between expert, knowledge engineer and user roles are deliberately blurred, and a diversity of knowledge processes and changing roles are supported within an entire interacting community. A prototype knowledge support system is described with examples of some of the knowledge acquisition and application tools provided. It is suggested that such systems provide a major knowledge-based technology with commercial implications and applications going beyond those currently envisioned for expert systems.

1. INTRODUCTION

Growing recognition of the industrial significance of knowledge-based computing systems has focused attention on processes of knowledge acquisition and transfer. Commercial application of expert systems is being impeded by the knowledge engineering bottleneck and has led to the development of rapid prototyping tools. The objective of systems such as PLANET (Shaw & Gaines 1983, 1986a), ETS (Boose 1984, 1985), MORE (Kahn, Nowlan, & McDermott 1985), SALT (Marcus, McDermott & Wang1985), KITTEN (Gaines & Shaw 1986b, Shaw & Gaines 1986b), AQUINAS (Boose 1986, Boose & Bradshaw 1987) and KSSO (Gaines 1987) is to expedite the process of knowledge acquisition and transfer to expert systems. These have proved practically useful but the range of applications of existing tools is limited, and it is not clear what limitations are inherent in the prototyping tools, what arise from the shells and their knowledge representation and inferencing procedures, and what arise from our lack of understanding of the underlying processes of knowledge acquisition and transfer. There has also been very little analysis of the developmental processes for knowledge-based system in relation to well-established techniques for system specification, rapid prototyping, evaluation, full-scale development, documentation and maintenance of conventional data-processing systems.

This paper analyzes the problems of rapid prototyping for expert systems, and develops the concept of automated knowledge acquisition systems, giving design objectives, some detailed design considerations, and examples of practical systems that have been implemented and applied to date.

2 CHANGING PARADIGMS IN EXPERT SYSTEM

The initial approach to the development of expert systems has been that shown in Figure 1 where an expert is interviewed by a knowledge engineer who encodes the elicited expertise in a knowledge base which is then used to provide expert advice to users through an expert system shell. It was possible that knowledge engineering might develop as a profession on a par with systems analysis and programming, and that an initial shortage of skilled knowledge engineers would cause problems to be overcome eventually be as the profession developed. However, this scenario now appears less and less likely. There is certainly a shortage of knowledge engineers and problems in developing applications, but doubts have been cast on the notion that human labor is the appropriate solution to the knowledge engineering problem:

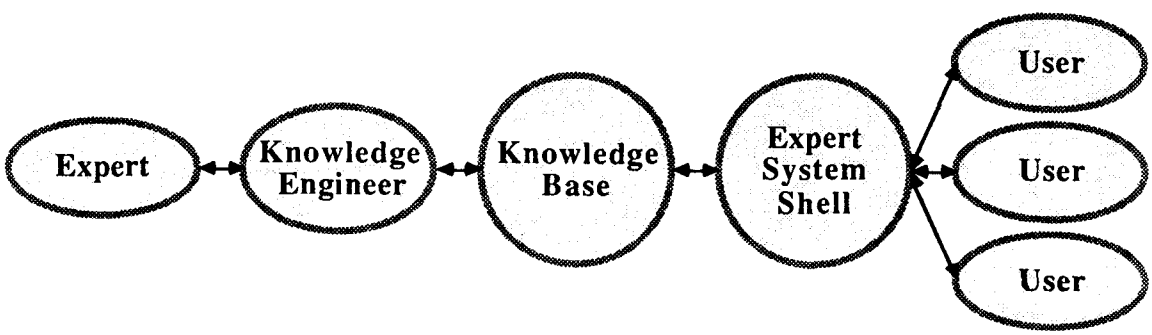

Fig. 1 Manual development of an expert system

© CIPS Edmonton 1987

- the decline in costs of both hardware and software support for expert systems has brought the technology into a mass-market situation far more rapidly than originally envisioned;
- this has lead to a growth in demand for expert systems that is proceeding far more rapidly than the growth in supply of trained and experienced knowledge engineers;
- the declining costs of expert system technology are also making the expense of human labor in tailoring the technology for particular applications appear to be the dominating constraint and an excessive cost;
- a move towards a labor-intensive activity such as knowledge engineering is contrary to all trends in industry;
- in particular it is contrary to the trend towards automatic programming techniques in the computing industry;
- the role of the knowledge engineer as an intermediary between the expert and the technology is being questioned not only on cost grounds but also in relation to its effectiveness—knowledge may be lost through the intermediary and the expert's lack of knowledge of the technology may be less of a detriment than the knowledge engineer's lack of domain knowledge.

The considerations of the previous section hav heightened interest in the possibility of providing interactiv knowledge acquisition systems to automate knowledg engineering as a process of direct interaction between exper and the computer. In 1980 we proposed that personal constru psychology (Kelly 1955, Shaw 1980) could provid foundations for expert systems, and gave examples o algorithms and programs that extracted inference rules fro repertory grid data (Gaines & Shaw 1980). In 1983 w reported further enhancements of these techniques and preliminary experiment to validate them empirically as knowledge engineering technique for priming expert systen (Shaw & Gaines 1983). This work led to industrial studies o the methodology applied to the development of expert system Boeing Computer Services (Boose 1984, 1985) and Lockhee Software Technology Center (Wahl 1986) have reporte success in applications; and validation has been reported in statistics domain (Gammack & Young 1985). Figure 2 show a family history of the development of interactive knowledg acquisition tools arising out of personal construct psychology.

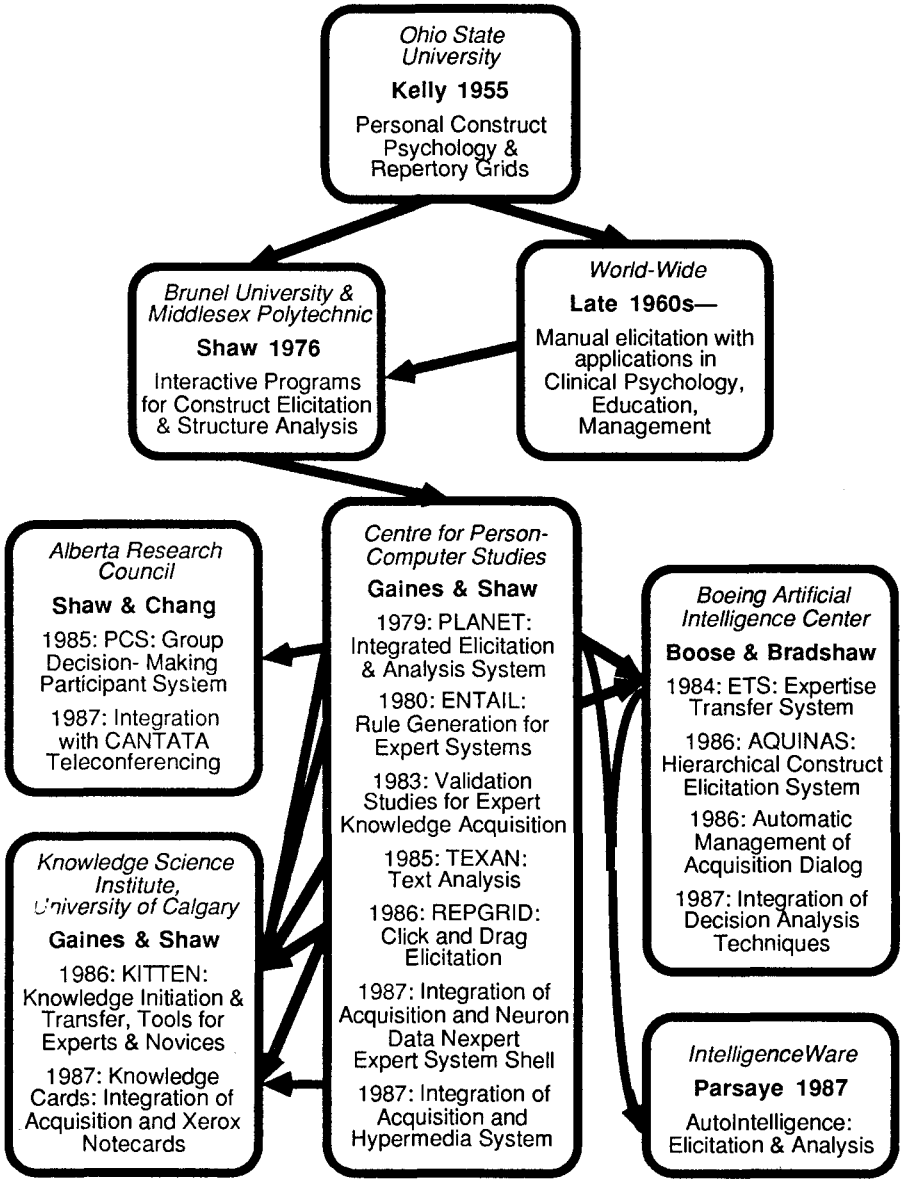

Fig. 2 Family tree of knowledge acquisition systems based on personal construct psychology

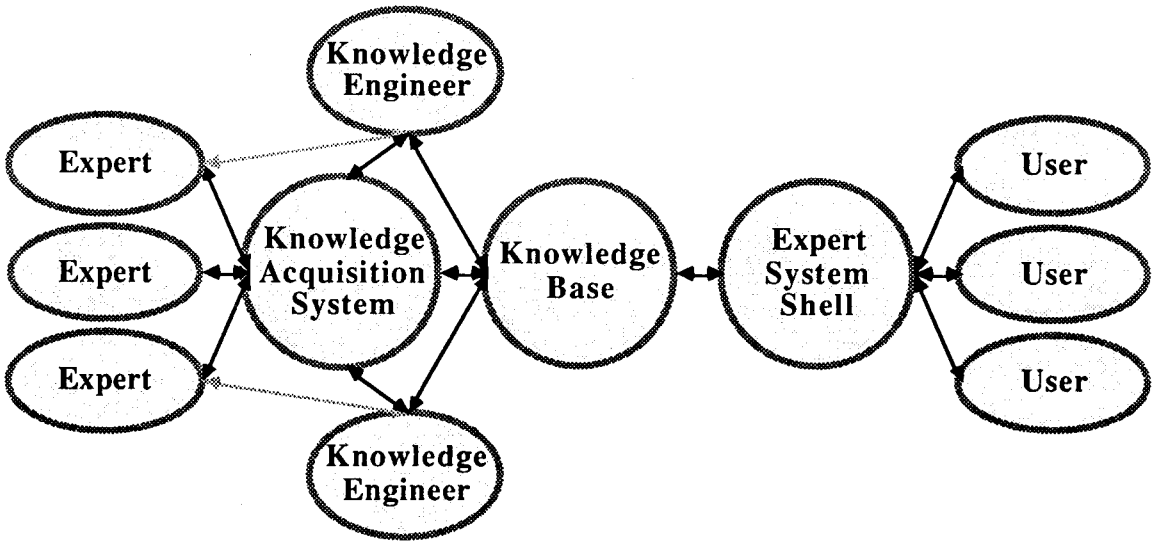

Fig.3 Automated knowledge acquisition for an expert system

Figure 3 shows how the role of the knowledge engineer changes with the application of interactive knowledge acquisition systems:

- The experts now interact directly with a knowledge acquisition system that aids them to develop a conceptual structure for their knowledge and express it directly in a computational representation.
- Knowledge engineers now act as managers helping the expert in the initial interaction, having overall control of the knowledge acquisition system and editing the resultant knowledge base. They now have a far lower workload and do not have the responsibilities of an intermediary.
- Multiple experts and multiple knowledge engineers are shown because the use of diverse sources of expertise has proved valuable in practice, and the automation of the knowledge acquisition function makes it possible to manage the complexities of multiple sources and their comparison.

The system shown in Figure 3 has a distinct break between acquisition and application. However, in practice the validation of the knowledge base usually depends on the expert and knowledge engineer becoming users and checking the advice given through the expert system shell against well defined case histories. It is useful for the role switching to be as simple as possible, and this has led to the increasing integration of the knowledge acquisition system with the expert system shell as noted in the previous section. It then becomes less and less meaningful to distinguish between 'experts', 'knowledge engineers' and 'users'.

Figure 4 shows the integration of the system with a central core of knowledge acquisition, knowledge base and expert system shell interacting with a community of users in diverse, mixed and varying roles. It is this that we have termed a *knowledge support system* because it potentially supports such a wide variety of knowledge processes. In particular, it allows the community to share knowledge in an explicit form, keep it updated from a variety of sources and use it flexibly, comparing and contrasting different perspectives.

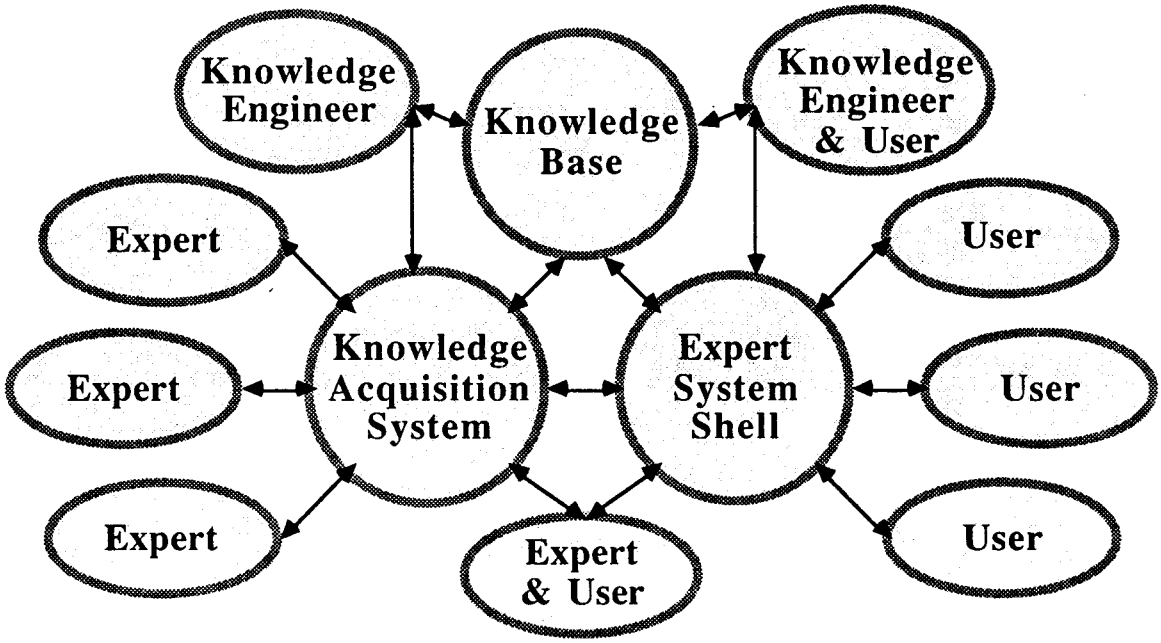

Fig.4 Integration of knowledge acquisition and expert system shell

The shell provides a number of facilities that interact with acquisition and these are shown explicitly in Figure 5 as part of the basic tools of a complete knowledge support system:

- *Knowledge base*: the central ring shows the knowledge base of facts and inference rules that is the heart of an expert system. This extends the facilities of a conventional database by enabling the storage not only of facts but also of rules that enable further facts to be derived. These rules may both be explicitly expressed (production rules) and implicit in the conceptual structure in which the facts are expressed (objects, frames).
- *Expert system shell*: the middle ring shows the expert system shell that provides computational facilities for applying the knowledge base to user decision support:

—the *application system* at the bottom tests the user's hypotheses or seeks to satisfy his goals by deriving the consequences of facts about a particular situation reported by the user, using the general facts and inference rules supplied by the expert and edited by the knowledge engineer;

—the *explanation system* to the lower left answers queries about the way in which facts have been derived in terms of what information and inference rules have been used;

—the *acquisition system* to the upper left provides tools for interviewing the expert to obtain his vocabulary, and general facts and inference procedures in terms of it;

—the *display system* at the top provides tools for presenting the knowledge base in an understandable form, and the relations between the facts and inference procedures it contains;

—the *edit system* to the upper right provides tools for editing the knowledge base while maintaining its integrity in terms of the vocabulary, variables and operations used;

—the *validation system* to the lower right provides tools for checking the knowledge base against specific case histories with known consequences.

- *Operators*: the outer ring shows the three classes of operator of expert systems:

—*users* come to the system for advice on specific problems in a particular domain, supplying it with specific facts and their hypotheses about consequences or goals;

—*experts* come to the system to transfer their knowledge about the problem domain, supplying it with generally applicable facts and inference procedures;

—*knowledge engineers* act as intermediaries between the expert and the system, helping him to encode his knowledge and validate the operation of the resultant expert system.

In considering system development, it is important to note that these operator roles may be combined within a single individual, such as the 'expert and knowledge engineer', making it dangerous to assume that three self-contained working environments can cater for the different levels of skill and conceptual frameworks involved. It is better to think in terms of a single integrated environment that allows different levels of access through mutually consistent operating procedures. Most current expert systems do not provide such an integrated environment, and complete modules are missing or inadequately implemented, but Figure 5 provides a framework within which all existing systems can be embedded.

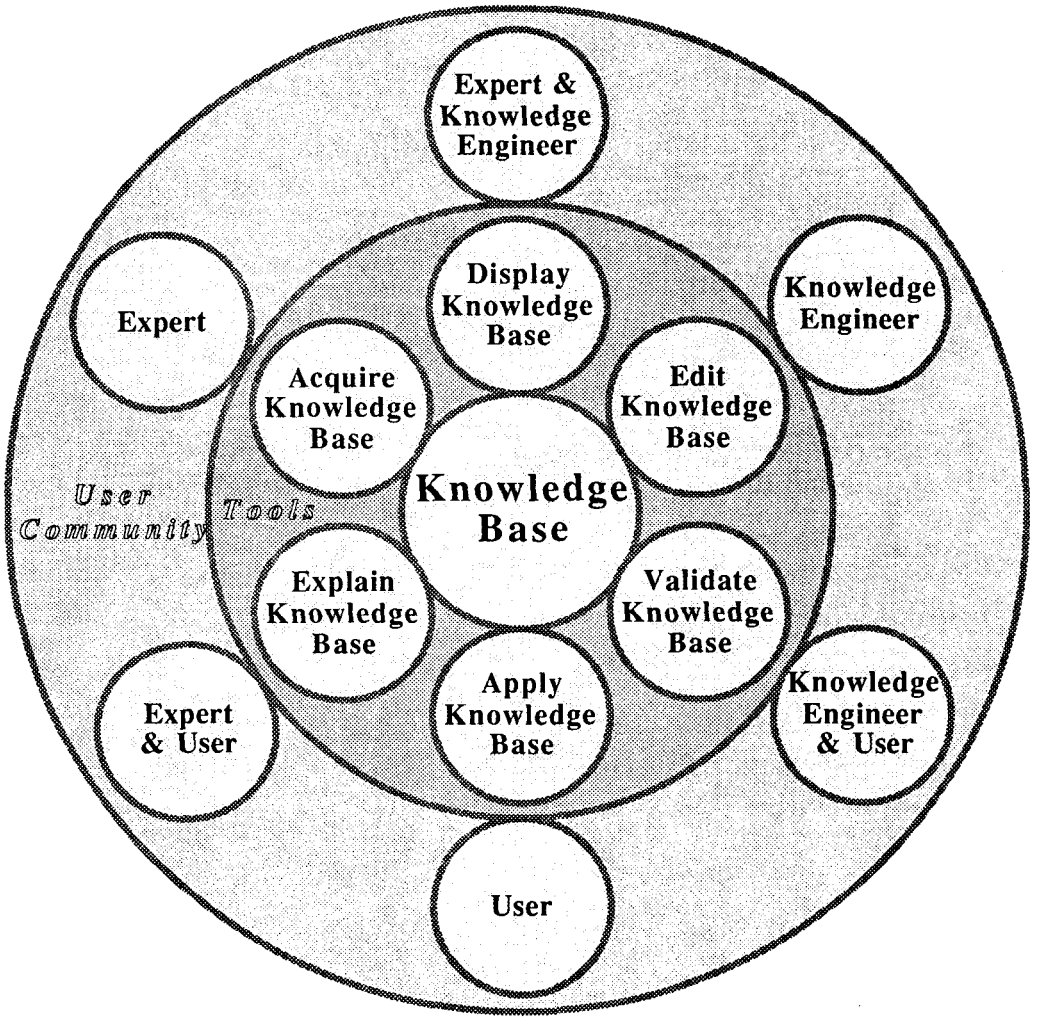

Fig.5 The architecture and operators of a knowledge support system

The shell provides a number of facilities that interact with acquisition and these are shown explicitly in Figure 5 as part of the basic tools of a complete knowledge support system:

- *Knowledge base*: the central ring shows the knowledge base of facts and inference rules that is the heart of an expert system. This extends the facilities of a conventional database by enabling the storage not only of facts but also of rules that enable further facts to be derived. These rules may both be explicitly expressed (production rules) and implicit in the conceptual structure in which the facts are expressed (objects, frames).
- *Expert system shell*: the middle ring shows the expert system shell that provides computational facilities for applying the knowledge base to user decision support:

—the *application system* at the bottom tests the user's hypotheses or seeks to satisfy his goals by deriving the consequences of facts about a particular situation reported by the user, using the general facts and inference rules supplied by the expert and edited by the knowledge engineer;

—the *explanation system* to the lower left answers queries about the way in which facts have been derived in terms of what information and inference rules have been used;

—the *acquisition system* to the upper left provides tools for interviewing the expert to obtain his vocabulary, and general facts and inference procedures in terms of it;

—the *display system* at the top provides tools for presenting the knowledge base in an understandable form, and the relations between the facts and inference procedures it contains;

—the *edit system* to the upper right provides tools for editing the knowledge base while maintaining its integrity in terms of the vocabulary, variables and operations used;

—the *validation system* to the lower right provides tools for checking the knowledge base against specific case histories with known consequences.

- *Operators*: the outer ring shows the three classes of operator of expert systems:

—*users* come to the system for advice on specific problems in a particular domain, supplying it with specific facts and their hypotheses about consequences or goals;

—*experts* come to the system to transfer their knowledge about the problem domain, supplying it with generally applicable facts and inference procedures;

—*knowledge engineers* act as intermediaries between the expert and the system, helping him to encode his knowledge and validate the operation of the resultant expert system.

In considering system development, it is important to note that these operator roles may be combined within a single individual, such as the 'expert and knowledge engineer', making it dangerous to assume that three self-contained working environments can cater for the different levels of skill and conceptual frameworks involved. It is better to think in terms of a single integrated environment that allows different levels of access through mutually consistent operating procedures. Most current expert systems do not provide such an integrated environment, and complete modules are missing or inadequately implemented, but Figure 5 provides a framework within which all existing systems can be embedded.

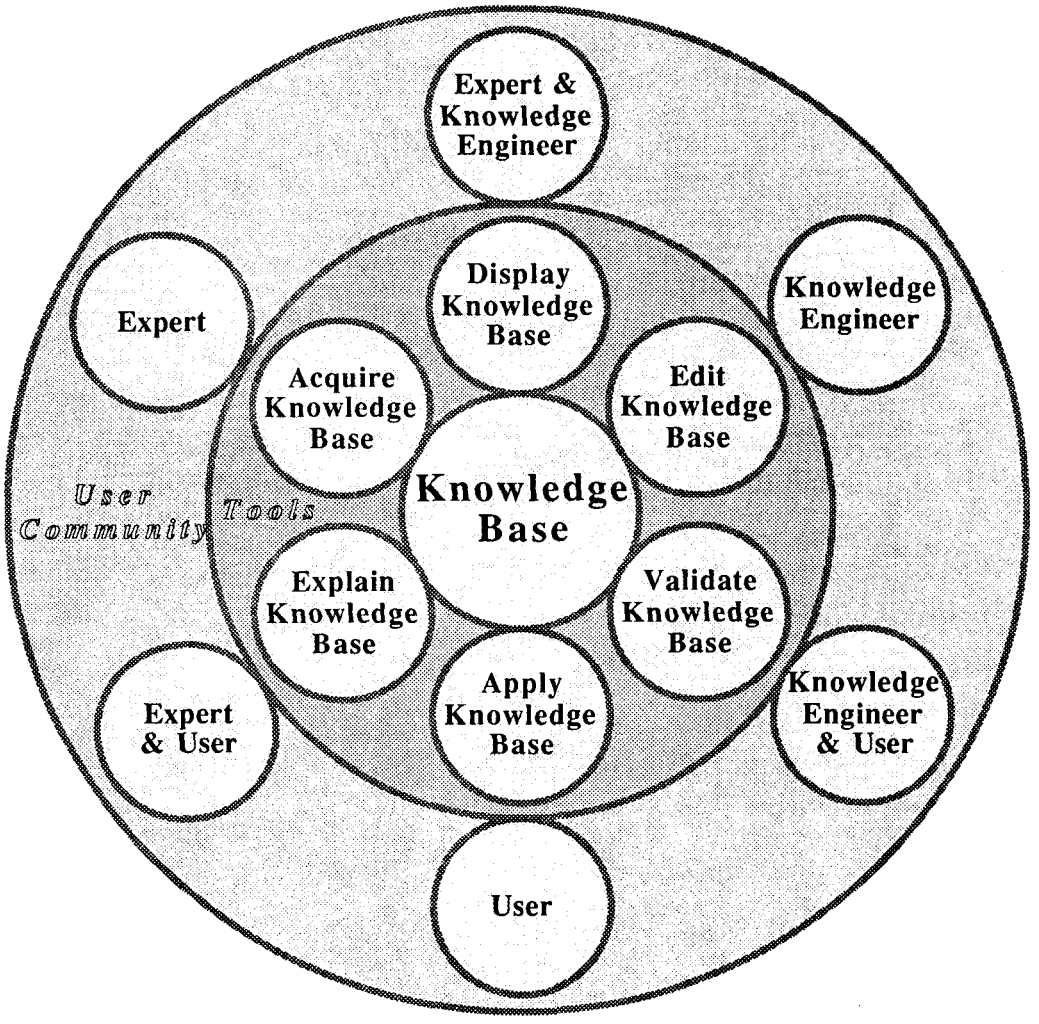

Fig.5 The architecture and operators of a knowledge support system

Most current expert system shells:

- offer a limited knowledge base represented in terms of production rules and frames;
- have weak application capabilities limited to deriving consequences and testing assertions;
- give simple explanations in terms of the facts and rules used;
- have no integrated acquisition capabilities;
- have limited capabilities to display the knowledge base, sometimes in graphical form;
- have interfaces to the knowledge base editor requiring understanding of internal data structures;
- have limited validation capabilities and no integrated database of case histories.

However, rapid improvements are taking place in the knowledge representation, inferencing, explanation, display and editing capabilities of expert systems (Hayes-Roth 1985, Michalski & Winston 1986, Pearl 1986, Swartout 1986, Rappaport 1986b). Most current knowledge acquisition is done by manual knowledge elicitation which places too much load on the knowledge engineer and inadequately involves the expert. Automatic tools for rapid prototyping are becoming increasingly important and are described in Section 6 with an example in Section 7.

3. THE VIRTUAL MACHINE HIERARCHY FOR EXPERT SYSTEM

One problem that system development managers often face in establishing prototyping capabilities for expert systems is the sheer variety of technologies apparently involved—should they use Lisp machines?—should they train their staff in Lisp or Prolog?—how does an expert system shell compare with existing programming languages?—what is knowledge engineering and how does it differ from systems analysis? The problem is that knowledge-based systems are at the top of the *virtual machine hierarchy* which has developed in computing, and there are many layers of technology between an expert system and the computer on which it is running. In theory, system developers should not need to know about the lower levels of the hierarchy—machine architectures, operating systems and implementation languages are remote from knowledge processing. However, in practice, these lower levels are the foundations on which systems are built and any defects in them can undermine the functionality of the upper levels.

Figure 6 shows significant levels of the virtual machine hierarchy for knowledge-based systems:

- *Knowledge Acquisition Tools*: at the top of the hierarchy are the tools for automating the systems analysis for expert systems, through automatic interview procedures, modeling actual expert behavior, and analysis of knowledge in textual form. Automation of systems analysis is more significant for expert systems than for conventional data processing because they tend to be targeted on narrow areas of in-depth specialist knowledge, rather than broad general problems. In conventional terms, expert systems are satisfying the requirement for highly customized data processing based on idiosyncratic operations rather than general algorithms.
- *Expert System Support Environment*: at the second level of the hierarchy is the equivalent of the Application Programming Support Environment (APSE) in conventional systems. The Expert System Support Environment (ESSE) has already been described in terms of Figure 5.
- *Expert System Shell*: at the third level is the shell as a run-time environment that elicits problem-specific information from the user, provides advice based on its knowledge base, and explains that advice in as much detail as required.
- *Expert System Shell Development Language*: at the fourth level is the language in which the expert system shell is written, generally a special-purpose environment for coping with knowledge representation and inference.
- *Implementation Language*: at the fifth level of the hierarchy is the implementation language which actually interfaces to the computer. This tended to be Lisp in the early days of expert systems but as speed and space efficiency have become significant, and knowledge representation has become better understood, other languages that support dynamic data structures such as C and Pascal have become widely used.
- *Operating System*: at the sixth level of the hierarchy is the operating system within which the implementation runs. This needs to provide good interfaces to other programs, large databases and communications.
- *Machine Architecture*: at the lowest level is the machine on which the expert system runs. The use of Lisp in the early days of expert systems has focused attention on Lisp machines. Nowadays any good workstation is adequate.

Fig.6 Hierarchy of virtual machines underlying expert systems

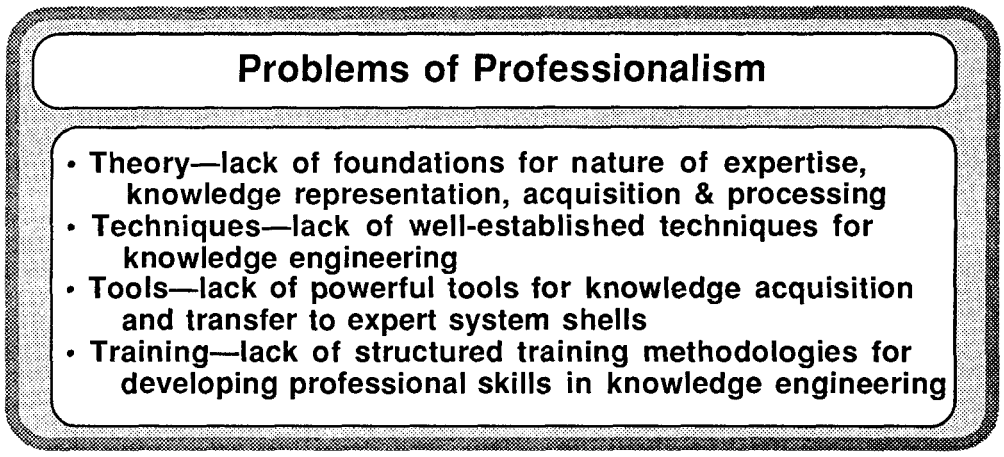

Fig.7 Problems of professionalism in developing expert systems

4. PROBLEMS IN ES DEVELOPMENT

The preceding section have given an overview of expert systems and an indication of some of the problems that may arise in their development. Many of these relate to the knowledge engineering bottleneck but there are many others that are not direct outcomes, and the knowledge engineering problems in themselves have many facets.

Figure 7 shows what may be termed the problems of *professionalism* in establishing prototyping facilities:

- *Theory*: the lack of scientific and engineering foundations for knowledge-based systems is an impediment to establishing professional standards for expert system development. We understand remarkably little about the nature of expertise even though it one of the most valued commodities in our society. We also have little fundamental understanding of knowledge, the ways in which it is represented in thought processes, the ways in which it is used in problem solving, and the ways in which it is transmitted between people.
- *Techniques*: the lack of well-established techniques for knowledge engineering is a major problem for new development teams. There is some folklore, a few rules of thumb based on experience, but so far no in-depth studies of what is actually done and how successful it has been.
- *Tools*: the lack of commercially available tools for knowledge acquisition and transfer is also a problem. There are now many research reports of tools but even those developed industrially have not yet been developed into products.
- *Training*: the training situation is improving all the time, largely because it has become the major secondary industry to the supply of expert system shells. A range of courses is being offered by commercial and university organizations. It is still a problem to get good quality courses targeted to specific requirements and on the schedules required.

Knowledge acquisition and transfer processes in human society are complex and poorly understood. Neurology, psychology, linguistics, education, sociology, anthropology, philosophy and systems theory all have contributions to make. However, integrating them, disambiguating jargon, and combining different objectives and perspectives are major problems. Figure 8 shows a variety of short-term problems arising from these weaknesses in foundations:

- *System Specification*: the formal specification of software that has made software engineering a professional discipline ensuring quality within well-defined budgets and schedules is lacking for expert systems. We are at the stage of conventional software development twenty years ago. This will change as expert system developments move out of laboratories into texperienced software development groups trained in software management. There is little in the literature to support and expedite this change currently, and we need to develop specification disciplines for expert systems.
- *Access to Expertise*: the problems of getting experts to make the time commitments necessary to the development of expert systems should never be underestimated. Successful projects inevitably seem to have involved the experts in such a way that they become personally committed to the success of the system.

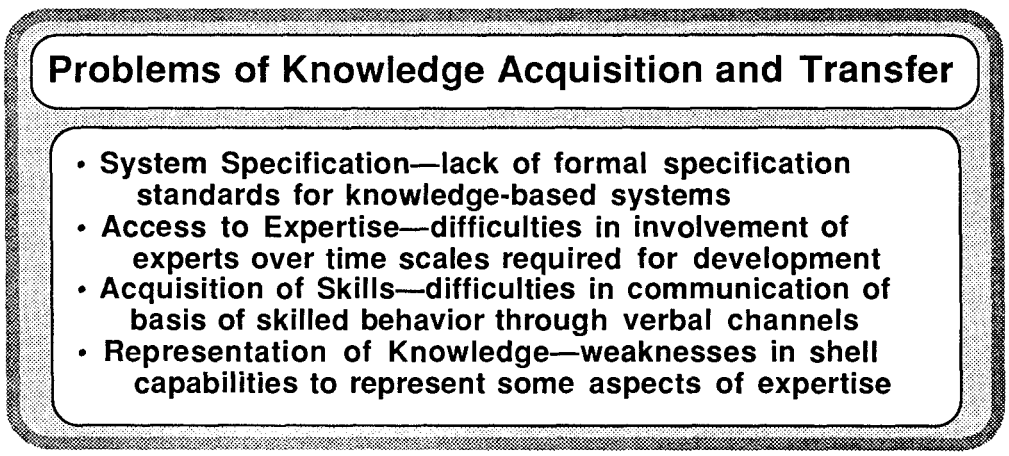

Fig.8 Problems of knowledge acquisition & transfer in developing expert systems

- *Acquisition of Skills*: communicating the basis of skilled performance between people, and between people and computers, is very difficult—this is analyzed in more detail in Section 5.
- *Representation of Knowledge*: encoding knowledge within the representation schemes of current shells can present major difficulties. Production rules as used in early expert systems offer very limited facilities for encoding simple knowledge structures at a single level. Frames and their generalization through objects allow conceptual hierarchies to be better expressed. Neither rules nor frames are adequate to encode the procedural or temporal knowledge which is often a key component of expertise.

Rapid prototyping is a significant practical objective and techniques for accelerated initial development without loss of long-term quality are now common in conventional data processing. However, expert system development emphasizes weaknesses in some areas of performance which we tend to take for granted in standard information technology. Figure 9 shows problems with:

- *Validation*: because we can rarely define and quantify expertise, the validation of expert systems can be very difficult. Do we subjectively validate against the opinions of the expert or do we attempt objective validation of performance against performance standards, accepting that the experts may.not themselves satisfy such criteria. How, do we determine whether the knowledge base captures the expert's relevant knowledge, accurately, consistently and completely? There are very few validation studies in the literature.
- *Usability*: the usability of expert systems raises many problems. If we accept the jargon of the expert and the rules of inference based on it, .how do we ensure that the operators will use this jargon in the same way as the expert so that the inferences remain valid. It is such considerations that make nonsense of the popular stereotype that access to encoded expertise can make experts of all of us. Understanding the questions, advice and explanations from an expert system generally requires an operator with the same professional background as the expert.
- *Maintainability*: the superficiality of the knowledge encoded in current expert systems raises major problems of maintenance. Keeping the knowledge base up to date may require an ongoing effort at the same level as the original development, a situation very different from most conventional data processing. The usual problems of extensive maintenance not introducing problems of its own are at least as severe for expert systems as for standard data processing.
- *Upgradability*: many expert systems, and the shells supporting them have been designed as stand-alone packages with little scope for integration into large-scale data processing environments. These systems will be difficult to upgrade and integrate with other software.

5. KNOWLEDGE ENGINEERING

The problem of knowledge acquisition for expert systems have already been emphasized as major developmental considerations and impediments to rapid prototyping. This is the domain of the knowledge engineer. Feigenbaum (1980) defined *knowledge engineering* as the reduction of a large body of knowledge to a precise set of facts and rules, and it has already become a major bottleneck impeding the application of expert systems in new domains. We need to understand more about the nature of expertise in itself and to able to apply this knowledge to the elicitation of expertise in specific domains.

The human factors problems of expertise elicitation from a skilled person are well-known in the literature of psychology (Nisbett & Wilson 1977, Broadbent, Fitzgerald and Broadbent 1986). Hawkins (1983) has analyzed the nature of expertise and emphasizes its severe limitations and dependence on critical assumptions which are often implicit. Dixon (1981) has surveyed studies showing that much human activity is not accessible to awareness. Collins (1985) has studied knowledge transfer processes among scientists and suggests that some knowledge may not be accessible through the expert, not only because he cannot express it, but also because he may not be aware of its significance to his activity. Bainbridge (1979, 1986) has reviewed the difficulties of verbal debriefing and notes that there is no necessary correlation between verbal reports and mental behavior, and that many psychologists feel strongly that verbal data are useless.

Clinical psychologists see the problem as one of cognitive defences that impede internal communication, and have developed techniques of verbal interaction to identify underlying cognitive processes (Freud 1914, Kelly 1955, Rogers 1967). These can be used to by-pass cognitive defences, including those resulting from automization of skilled behavior. Welbank (1983) has surveyed the psychological problems of doing this.

The main problems identified in accessing an expert's knowledge are (Gaines 1986):

- "Expertise" may be fortuitous. Results obtained may be dependent on features of the situation which the expert is not controlling.

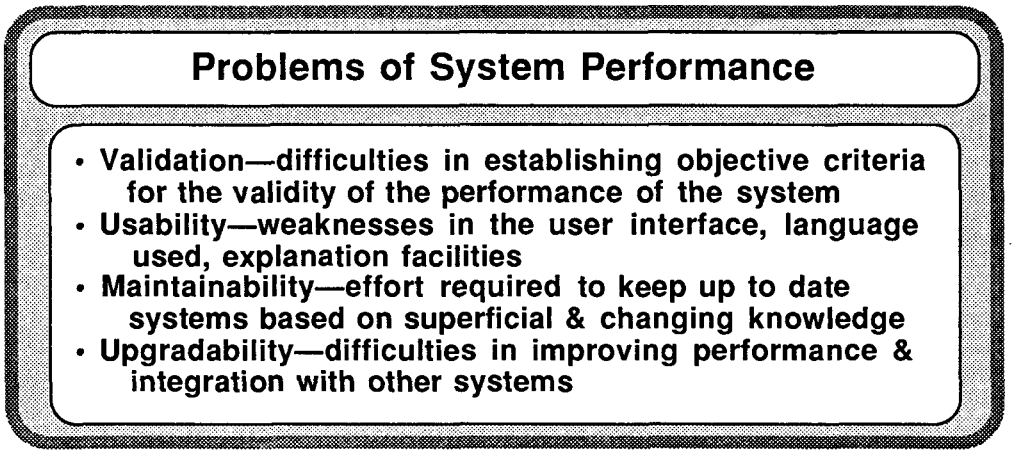

Fig.9 Problems of system performance in developing expert systems

- Expertise may not be available to awareness. Experts may be unable to transmit expertise by critiquing the performance of others because they are unable to evaluate it.
- Expertise may not be expressible in language. Experts may be unable to transmit expertise explicitly because they are unable to express it.
- Expertise may not be understandable when expressed in language. An apprentice may not be able to understand the language in which the expertise is expressed.
- Expertise may not be applicable even when expressed in language. An apprentice may not be able to convert verbal comprehension of ta skill into skilled performance.
- "Expertise" expressed may be irrelevant. Much of what is learnt, particularly under random reinforcement schedules, is superstitious behavior that neither contributes nor detracts from performance.
- Expertise expressed may incomplete. There will usually be implicit situational dependencies that make explicit expertise inadequate for performance.
- "Expertise" expressed may incorrect. Experts may make explicit statements which do not correspond to their actual behavior and lead to incorrect performance.

In the development of rapid prototyping techniques for expert systems the emphasis has been on interviewing experts, and hence on linguistic transmission of expertise. A rich variety of alternative methods for expertise transfer exists in human society:

- Expertise may be transmitted by managing the learning environment. Trainers may be able to establish conditions for an apprentice to acquire expertise without necessarily understanding the skill or themselves being expert in it.
- Expertise may be transmitted by evaluation. Trainers may be able to induce expertise by indicating correct and incorrect behavior without necessarily understanding the skill in detail or themselves being expert in its performance.
- Expertise may be transmitted by example. Experts may be able to transmit a skill by showing their own performance without understanding the basis of the expertise.

There are many sources of expertise that do not involve others:

- Expertise may be acquired by trial and error learning. This is the basic inductive knowledge acquisition process that is always in operation although heavily overlaid by the social transfer processes already discussed.
- Expertise may be acquired by analogical reasoning. The transfer of models and skills from one situation to another is an important source of knowledge.
- Expertise may be acquired by the application of general laws and principles to new situations. The use of physical laws and systemic principles to generate specific expertise is the basis of scientific and engineering expertise.

These many aspects of the problems of knowledge acquisition and transfer are further confounded by the combinatorial possibilities resulting from the recursive nature of knowledge processes. We can acquire knowledge about knowledge acquisition. We can mimic the behavior of an expert coach managing a learning environment. We can give evaluative feedback on analogical reasoning or the application of principles and laws. We can express the principles behind the effective management of learning in specific domains. The knowledge processes of human society are rich and complex, and, in practice, many of these possibilities will be instantiated as parallel, inter-related activities in any knowledge acquisition and transfer situation.

These considerations have heightened interest in the possibility of providing knowledge support systems to automate knowledge engineering as a process of direct interaction between domain experts and the computer. A brief review of such developments is given in the next section.

6. AUTOMATING KNOWLEDGE ACQUISITION

The problems outlined above are not unreasonable in the initial stages of a new technology. However, this makes it very important to understand to what extent expert systems do actually represent a new technology. If we understand the similarities and differences between past system design and expert system design then we can used well-established techniques in similar areas, and begin to develop effective techniques for managing the differences. It is clear that expert systems use some new computational techniques, but perhaps less apparent that the system design approach involved is itself fundamentally changed. However, major paradigm shifts are involved (Gaines & Shaw 1985).

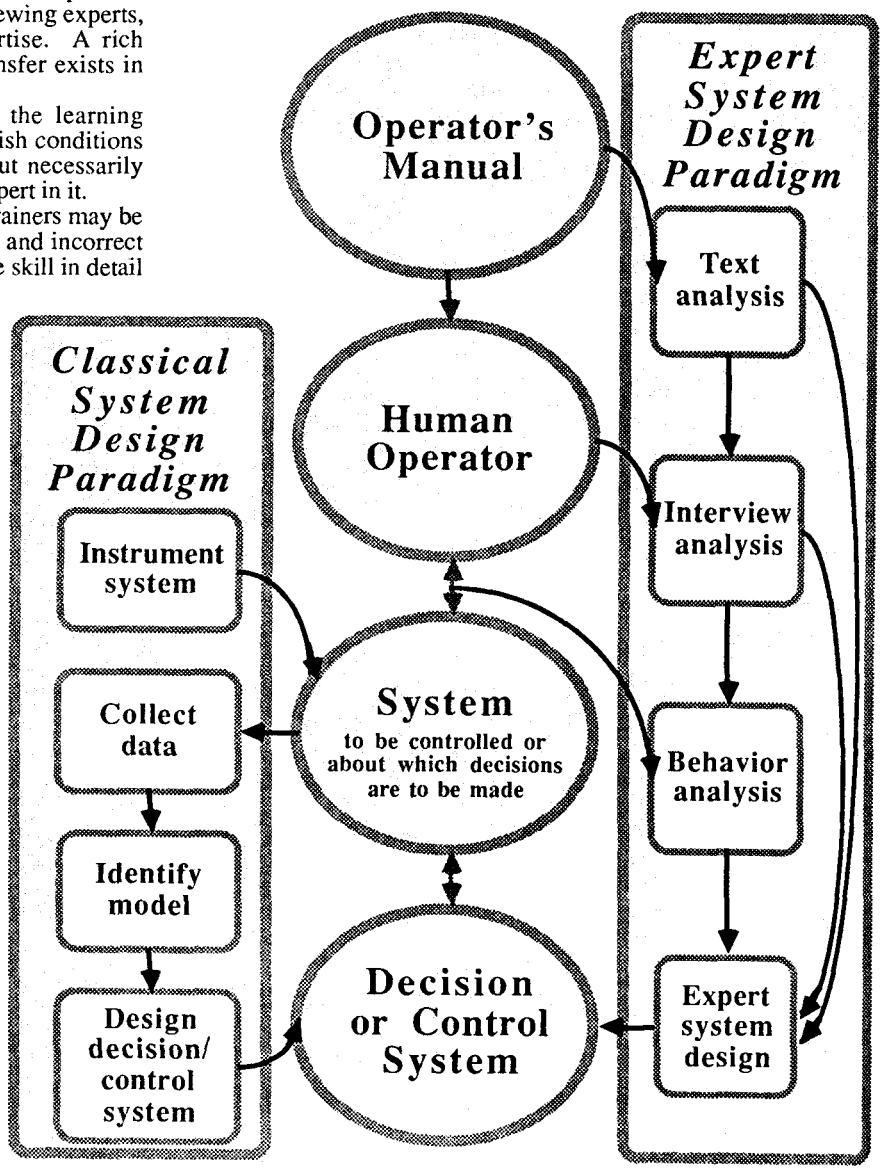

Fig.10 The expert systems approach to system development

The classical approach in decision and control system design is the instrumentation, data collection, modeling and optimization sequence shown on the left hand side of Figure 10. This approach to system design underlies the methodologies of the physical sciences and technologies based on them. It has the merit that it has been extremely successful in engineering much of the technological infrastructure of our current society including our manufacturing industries.

The right hand side of Figure 10 shows the classical system paradigm extended to include the derivation of design information from the human operator. The obvious approach is *interview analysis*, and this has been used in many studies. However, often the operator is unable to give a verbal description of his skill, or suggests an incorrect basis for it, and *behavior analysis* through computer modeling of his input-output behavior gives better results (Michalski, & Chilausky 1980). *Text analysis* of the operator's handbook is another significant source of information about his intended behavior.

A number of systems have been developed to automate the expert system design paradigm shown on the right of Figure 10 by interactively conducting a structured interview with experts, eliciting the experts' behaviors and then inductively modeling them, or by analyzing textual material. No one approach is adequate to capture all aspects of expertise and recent systems combine all three techniques. The next generation of knowledge acquisition systems can be expected to integrate the development processes of both sides of this diagram.

The general requirements for a system to automate knowledge acquisition for expert systems are:

- The knowledge acquisition system tools should be domain independent;
- The tools should be directly applicable by experts without intermediaries;
- The tools should be able to access a diversity of knowledge sources including text, interviews with experts, and observations of expert behavior;
- The knowledge acquisition system should be able to encompass a diversity of perspectives including partial or contradictory input from different experts;
- The system should be able to encompass a diversity of forms of knowledge and relationships between knowledge;
- The system should be able to present knowledge from a diversity of sources with clarity as to its derivation, consequences and structural relations;
- Users of the knowledge acquisition system should be able to apply the knowledge in a variety of familiar domains and freely experiment with its implications;
- The system should make provision for algorithmic expression of knowledge where appropriate;
- The system should make provision for validation studies;
- As much of the operation of the system as possible should be founded on well-developed and explicit theories of knowledge acquisition, elicitation and representation;
- As the overall knowledge acquisition develops it should converge to an integrated system.

All of these requirements are subject to caveats—some domain dependence may be appropriate for efficiency in specific knowledge acquisition systems—some human intervention may be helpful or necessary when an expert is using the system—and so on. However, the broad design goals stated capture the key issues in the knowledge acquisition system design community.

The next section briefly describes KSS0, a fifth generation development of such techniques as an integrated suite of knowledge acquisition programs.

7. KNOWLEDGE SUPPORT SYSTEM ZERO

Hayes-Roth, Waterman & Lenat (1983) have analyzed the way in which communication problems impede the process of transferring expertise into a program. They note that he vocabulary initially used by the expert to talk about the domain with a novice is often inadequate for problem-solving, so that the knowledge engineer and expert must work together to extend and refine it. One of the most difficult aspects of the knowledge engineer's task is helping the expert to structure the domain knowledge, to identify and formalize the domain concepts. This elicitation of an adequate terminology, and its structuring, are the primary functions of prototyping tools based on repertory grid techniques.

The basic concepts underlying repertory grids are those already familiar in computing of *entity-attribute* databases. The expert is asked to define classes of entities that are relevant to different areas of his problem domain, and then to derive significant attributes of these entities by considering how he makes distinctions between them. This process is best seen through some practical examples.

Figure 11 is the startup screen for KSS0. The system is designed to be very clean and simple to use with a substantial entertainment value through the use of digital speech and animation. The objective is to make interaction with system attractive and fun while retaining a professional working environment. Spoken and visual help are available at all times and context sensitive manual sections are stored on the bottom of each screen so that they are accessible through scrolling. The expert should never be at a loss, need help he cannot get, or have to access a manual.

The default action at all times is to click on the button at the top right of the lower control panel—usually "OK", "Yes", "More" or the most likely action. During elicitation, when the user has a choice of actions, this default action will select one for him by biassed random choice from a reasonable set generated by rules which can be edited by the knowledge engineer. Thus, experts can begin to make effective use of the system very quickly with no need for computer experience and very little hand holding.

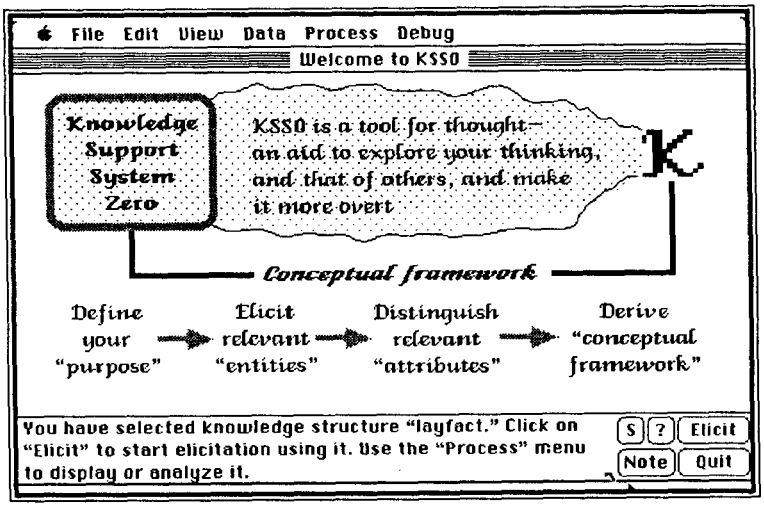

Figure 11 KSS0 startup screen showing process of elicitation

Figure 12 is an entity screen from KSS0 in a knowledge elicitation sequence concerned with factory layout planning (Adler & Matheson 1986). KSS0 allows its vocabulary to be tailored to the problem to maximize expert involvement, and the expert and knowledge engineer have agreed on a basic terminology. The expert has entered various "factors" which affect layout—these are the entities, and we will see the system help him to elicit relevant "qualities"—these are the attributes.

The system keeps track of matches between entities and between attributes which indicate structures in the knowledge, and the expert has decided to investigate one of these. The indicator at the lower left of Figure 11 shows the value of the highest match between entities. The expert has clicked on it to highlight the matching entities. He can use the "-" and "+" buttons to step backwards and forwards through the matches.

He next clicks on the "Break" button to see whether he can input an attribute that will reduce the match—that is, a counter-example to the inferred structure. KSS0 takes him to the screen shown in Figure 13 which makes explicit the question he has to answer and shows him how the match has come about. The horizontal bars show the ratings of the entities on the attributes, with those of the upper entity ("location of services") being shown by the upward markers, and those of the lower entity ("area of department") being shown by the downward markers.

The expert can edit these ratings by dragging the markers along the scales, or modify the entity names by clicking on them to open up a text edit window. At no time in KSS0 is he constrained to follow a specific course of action since the system is designed to facilitate free exploration and development of knowledge structures. In this case the expert decides to complete his action by answering "OK" to the question asked. KSS0 then takes him to the screen shown in Figure 14 where he is prompted to define a new attribute and then rate all the entities on it by clicking and dragging them to the vertical rating bar in the center.

Figure 15 shows the start of the complementary sequence in which the expert starts from the attribute screen and clicks on the indicator to see the highest matched attributes. He then clicks on the "Break" button again and moves to the screen shown in Figure 16 where the basis of the match is shown and again a very specific question is asked about the type of entity required to break the match. He can again edit the ratings by clicking and dragging the entities. Answering "OK" to this question takes him to screen 17 where he can enter the name of the new entity and then rate it on each of the existing attributes.

Figure 18 shows a different form of attribute elicitation initiated when the expert clicks on the "Triple" button in Figure 12. KSS0 generates a triple of three entities and asks him in what way two are alike and differ from the third. He clicks on "worker amenities" and KSS0 takes him to the rating screen shown in Figure 19 with the triple pre-rated, so that he may enter the attribute and the ratings of the other entities. The triples are generated on a random basis biassed to promote variety, and this mode of elicitation tends to trigger recollections of significant attributes otherwise forgotten.

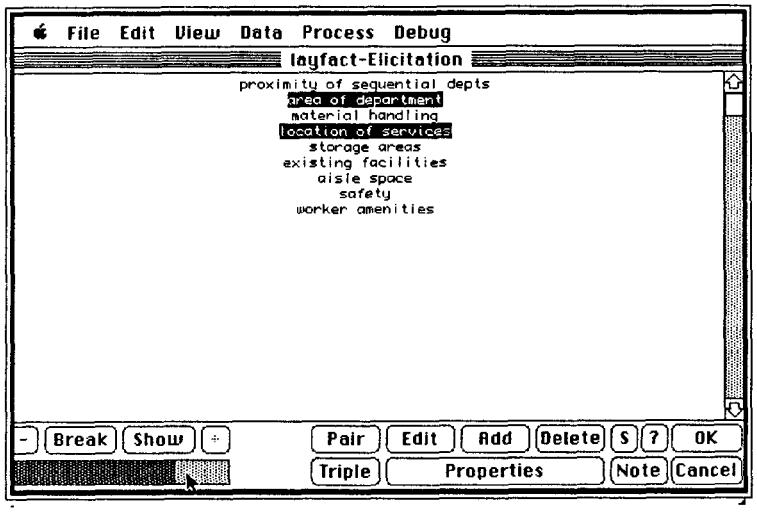

Figure 12 KSS0 entity screen showing matching entities

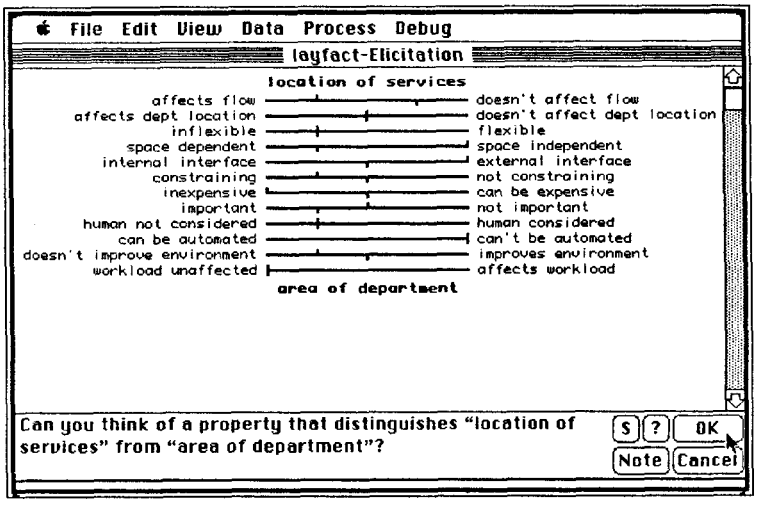

Figure 13 KSS0 entity match screen prompting new attribute

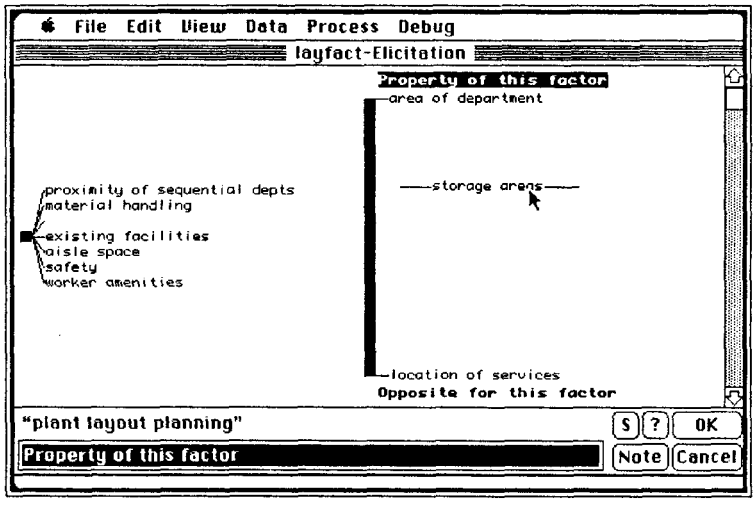

Figure 14 KSS0 attribute entry and rating screen

At any time during the elicitation the expert can ask for the structure of his entity-attribute conceptual system to be analyzed. Figure 20 shows a hierarchical clustering of his entities and attributes based on the matches between them. Figure 21 shows an alternative mapping into two-dimensions based on a principal components analysis. Figure 22 shows a third form of analysis based the logical *entailments* in his data (Gaines & Shaw 1980, 1986a). This third form is used to generate rules with which to prime an expert system shell.

Figure 23 shows two of the clustering analyses being used in the middle of an elicitation to see the way in which a conceptual structure is developing. Figure 24 shows the logical entailments being used within the expert system shell, *Nexpert* (Rappaport 1986a,b) in a prototype expert system based on the conceptual structures whose elicitation has been shown together with rules relating these concepts to layout decisions.

The technical details of the processes going on in Figures 11 through 24 are less important than the psychological impact on the expert. Users of KSS0 generally find that it gives fascinating insights into their own thought processes which both makes it attractive and puts them into an introspective frame of mind conducive to knowledge elicitation. Most of the dialog they see on the screen is generated from their own inputs and hence appears natural in terms of their professional terminology. Most importantly, this translates immediately into the rules generated for use in the expert system shell—they use the expert's jargon and their derivation from the data entered has already been shown to the expert and changes negotiated.

Thus, the initial prototype system, such as that shown in Figure 24, is comprehensible to the expert in its derivation and terminology. Also, by that stage the expert has become accustomed the use of the computer, and will often move over to editing the rules in the shell without really noticing that the environment has changed.

In most problem environments there are multiple knowledge islands, each requiring a separate elicitation process specific to the knowledge structures within the island. These tend to form a hierarchy, for example "safety" is a general factor in the elicitation of Figure 12 but has its own detailed conceptual structure in terms of a variety of hazards, a variety of forms of accident, a variety of legal implications, and so on. The methodology shown above can be extended to deal with such hierarchies (Boose & Bradshaw 1987).

KSS0 also contains facilities for extracting conceptual clusters by analyzing word associations in textual material. This is useful for defining knowledge islands, and pump-priming the related elicitations with entities and attributes. The logical entailment analysis in KSS0 is a rule induction process that can be applied to examples of expert decisions encoded in entity-attribute form. Thus the three forms of expert system design shown in Figure 10 are incorporated in an integrated form in KSS0.

8. SUMMARY AND CONCLUSIONS

This paper has spent at least as much space on the problems creating major difficulties for expert system development as it has on techniques for rapid prototyping. However, identification of the problems in

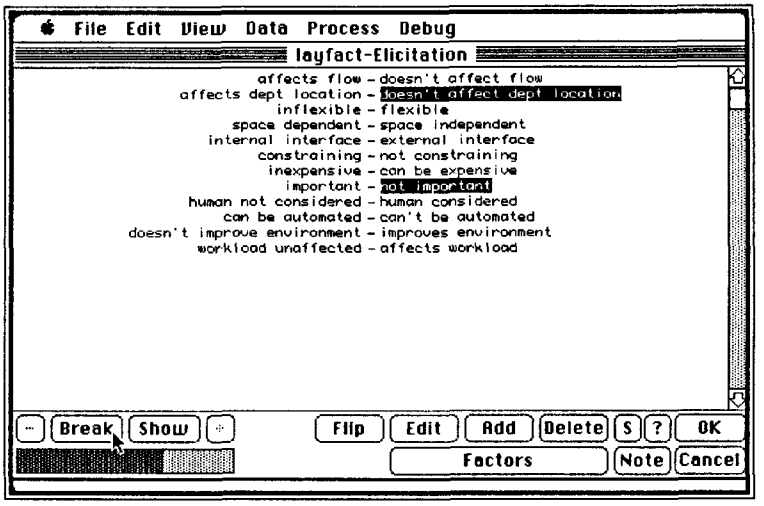

Figure 15 KSS0 attribute screen showing matching attributes

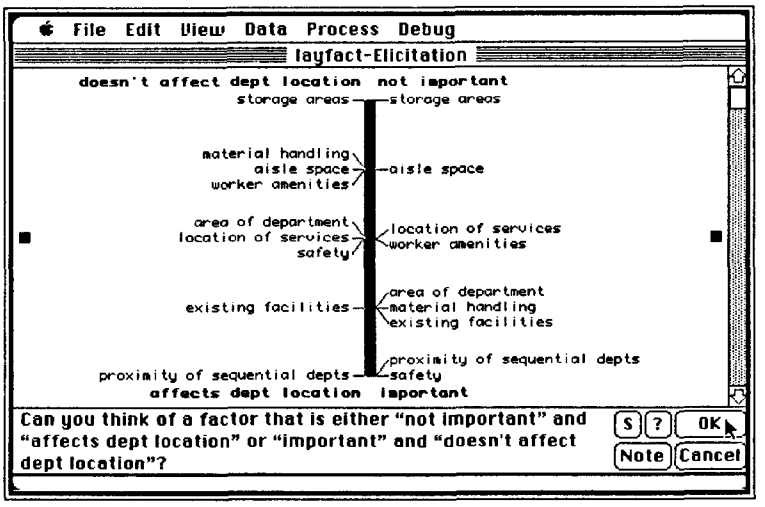

Figure 16 KSS0 attribute match screen prompting new entity

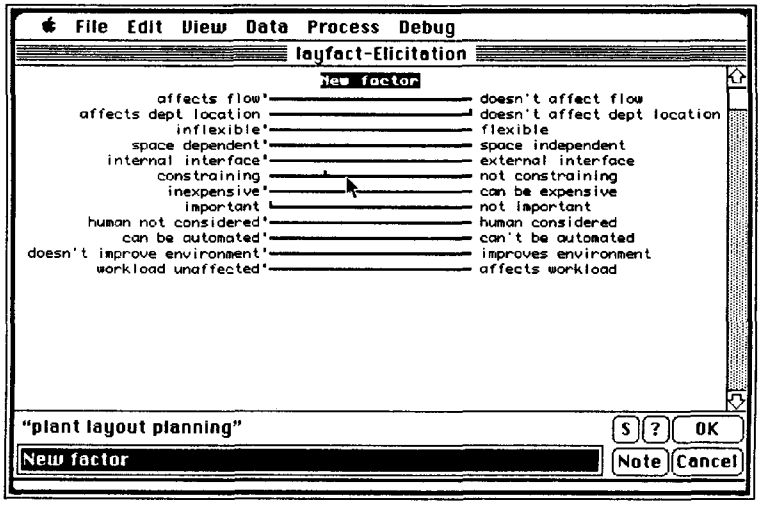

Figure 17 KSS0 entity entry and rating screen

system development is the key to overcoming those problems in practice, or, at least, to preparing proposals, budgets, schedules and development programs taking into account the problems likely to be encountered. We have to move beyond the stage where knowledge engineering has been identified as a bottleneck to one where the detailed architecture of that bottleneck is well-defined. We have to be able to classify the experience of both successes and failures in expert system development so that we can learn from them.

The main message in in Figure 7—the four "T's" of theory, techniques, tools and training are the most significant to practical expert system development. Psychological theories of knowledge processes in people, techniques for structured interviews, tools such as KSS0, and training in the application of all three to practical problems—these are the foundations of knowledge engineering.

Rapid prototyping is particularly important because it gives short-term feedback to the expert. If there is one lesson that can be drawn from most reported studies in successful knowledge engineering, it is "involve the expert." We have already moved into an era of end user computing, and we are now moving into an era of end user development—intermediaries are out. This is necessary not just because of distortions introduced by intermediaries, but also because most expert systems will involve very high levels of maintenance activity to keep the knowledge base up to date. Experts will find this an increasing professional responsibility, and need the best possible support tools in doing this. We can expect increasing integration between knowledge acquisition tools such as KSS0 and expert system shells such as Nexpert, and decreasing differentiation between acquisition, maintenance and application.

9. ACKNOWLEDGEMENTS

Financial assistance for a substantial part of this work has been made available by the National Sciences and Engineering Research Council of Canada. Many of the developments described result from joint research with Mildred Shaw. I am grateful to my colleagues, John Boose, Jeff Bradshaw and Mildred Shaw for access to their own work and stimulating discussions.

10. REFERENCES

Adler, J. & Matheson, J. (1986). Expert knowledge elicitation using personal construct theory: obtaining plant layout expertise with 'PLANET'. **BASc Thesis**. Department of Industrial Engineering, University of Toronto (April).

Bainbridge, L. (1979). Verbal reports as evidence of the process operator's knowledge. **International Journal of Man-Machine Studies, 11**(4), 411-436 (July).

Bainbridge, L. (1986). Asking questions and accessing knowledge. **Future Computing Systems 1**(2), 143-149.

Boose, J.H. (1984). Personal construct theory and the transfer of human expertise. **Proceedings AAAI-84**, 27-33. California: American Association for Artificial Intelligence.

Boose, J.H. (1985). A knowledge acquisition program for expert systems based on personal construct psychology. **International Journal of Man-Machine Studies 20**(1), 21-43 (January).

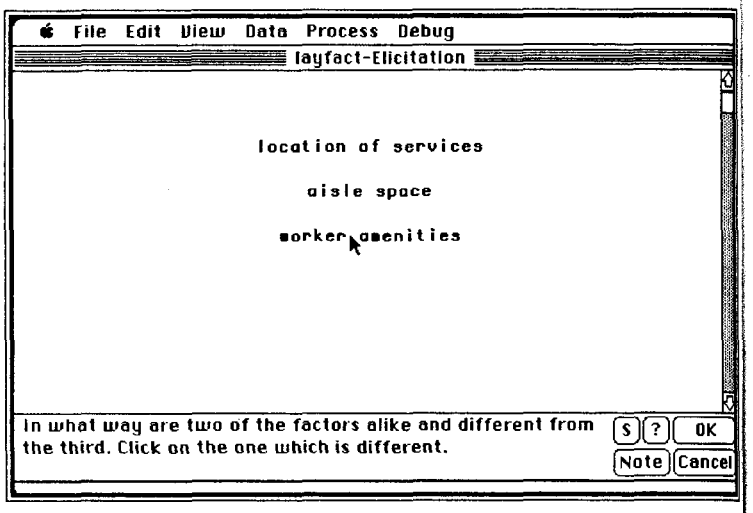

Figure 18 KSS0 attribute elicitation from a triple of entities

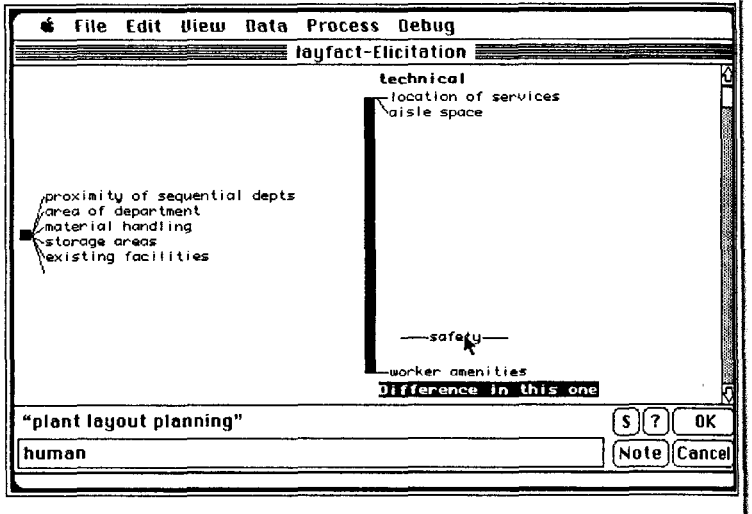

Figure 19 KSS0 attribute entry and rating screen after triple

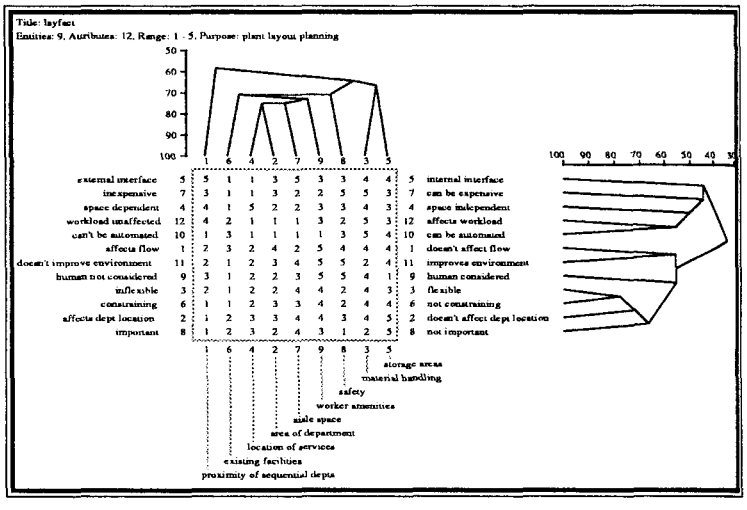

Figure 20 KSS0 hierarchical clustering of entities and attributes

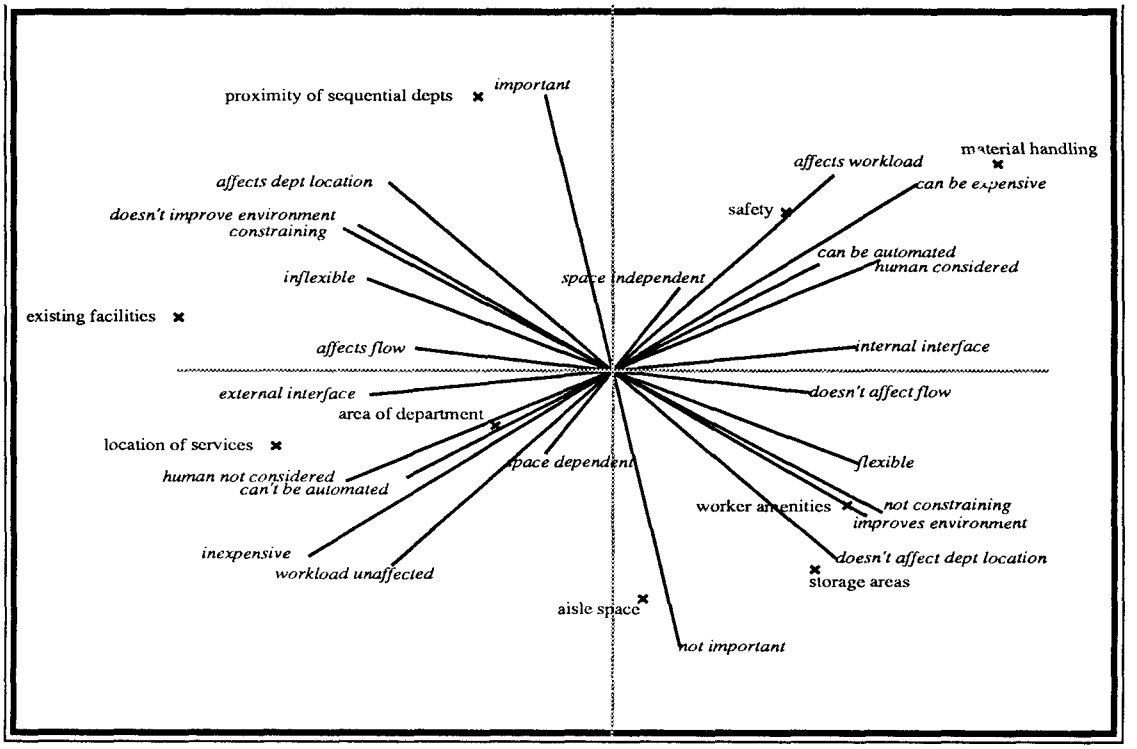

Figure 21 KSS0 principal components map of entities and attributes

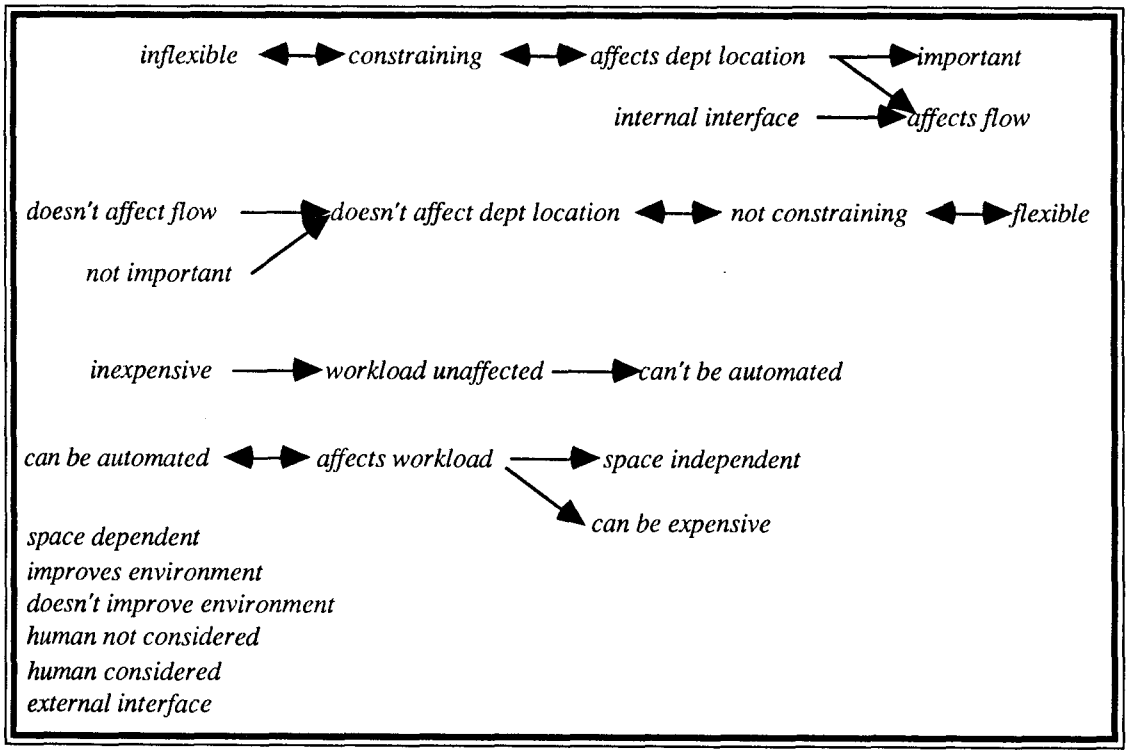

Figure 22 KSS0 logical analysis of entailments

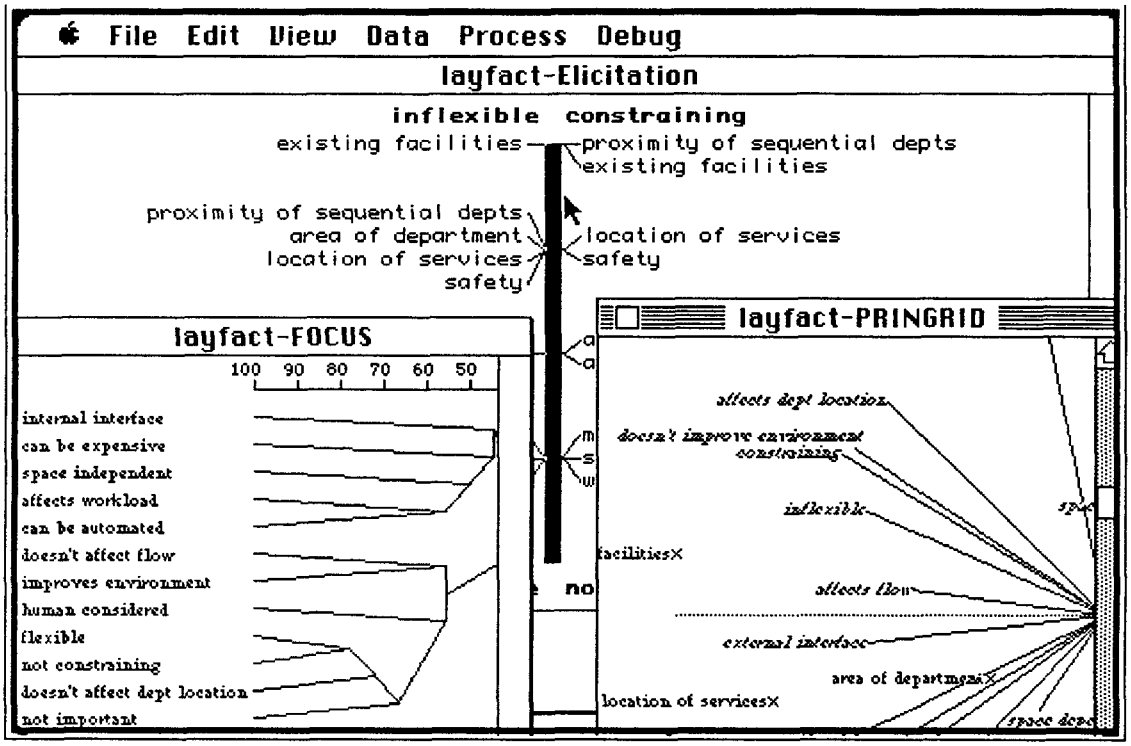

Figure 23 KSS0 access to clustering during elicitation

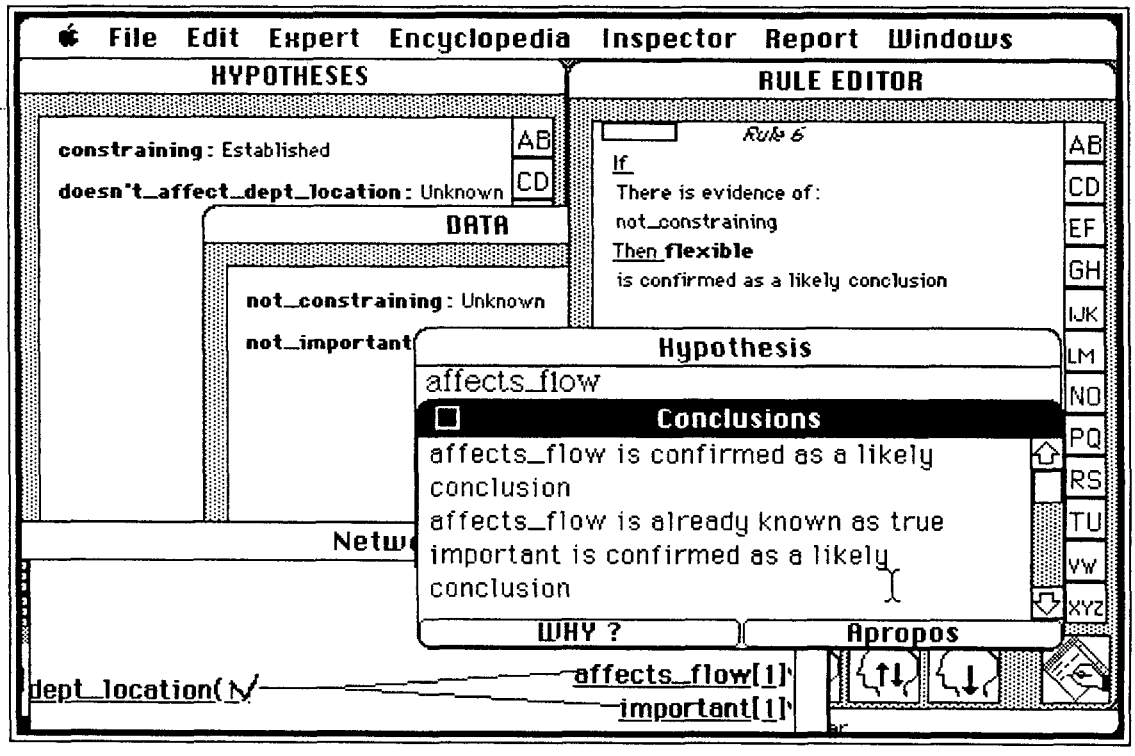

Figure 24 Nexpert operating with rules generated by KSS0

Boose, J.H. (1986). Rapid acquisition and combination of knowledge from multiple experts in the same domain. **Future Computing Systems 1**(2), 191-216.

Boose, J.H. & Bradshaw, J.M. (1987). Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for expert systems. **International Journal of Man-Machine Studies 26**(1) (January).

Broadbent, D.E., FitzGerald, P. & Broadbent, M.H.P. (1986). Implicit and explicit knowledge in the control of complex systems. **British Journal of Psychology 77**, 33-50.

Collins, H.M. (1985). **Changing Order: Replication and Induction in Scientific Practice**. London: SAGE.

Dixon, N. (1981). **Preconscious Processing**. Chichester: Wiley.

Feigenbaum, E.A. (1980). Knowledge Engineering: the Applied Side of Artificial Intelligence. **Report STAN-CS-80-812**. Department of Computer Science, Stanford University.

Freud, S. (1914). **Psychopathology of Everyday Life**. London: Benn.

Gaines, B.R. (1986). Foundations of knowledge engineering. Bramer, M.A., Ed. **Research and Development in Expert Systems III**. pp.13-24. Cambridge: University Press.

Gaines, B.R. (1987). Rapid prototyping for expert systems. Oliff, M., Ed. **Proceedings of International Conference on Expert Systems and the Leading Edge in Production Planning and Control**. pp.213-241. University of South Carolina.

Gaines, B.R. & Shaw, M.L.G. (1985). From fuzzy sets to expert systems. **Information Science 36** (1-2), 5-16 (July).

Gaines, B.R. & Shaw, M.L.G. (1980). New directions in the analysis and interactive elicitation of personal construct systems. **International Journal of Man-Machine Studies 13**(1) 81-116 (July).

Gaines, B.R. & Shaw, M.L.G. (1986a). Induction of inference rules for expert systems. **Fuzzy Sets and Systems**, 8(3), 315-328 (April).

Gaines, B.R. & Shaw, M.L.G. (1986b). Knowledge engineering techniques. **Proceedings of AUTOFACT'86**. 8-79-8-96. Detroit: Society of Manufacturing Engineers (November).

Gammack, J.G. & Young, R.M. (1985). Psychological techniques for eliciting expert knowledge. Bramer, M., Ed. **Research and Development in Expert Systems**, pp. 105-116. Cambridge University Press.

Hawkins, D. (1983). An analysis of expert thinking. **International Journal of Man-Machine Studies**, **18**(1), 1-47 (January).

Hayes-Roth, B. (1985). A blackboard architecture for control. **Artificial Intelligence**, **26**(3), 251-321 (July).

Hayes-Roth, F., Waterman, D.A. & Lenat, D.B., Eds. (1983). **Building Expert Systems**. Reading, Massachusetts: Addison-Wesley.

Kahn, G., Nowlan, S. & McDermott, J. (1985). MORE: an intelligent knowledge acquisition tool. **Proceedings of the Ninth International Joint Conference on Artificial Intelligence**. pp. 581-584. California: Morgan Kaufmann.

Kelly, G.A. (1955). **The Psychology of Personal Constructs**. New York: Norton.

Marcus, S., McDermott, J. & Wang, T. (1985). Knowledge acquisition for constructive systems. **Proceedings of the Ninth International Joint Conference on Artificial Intelligence**. pp. 637-639. California: Morgan Kaufmann.

Michalski, R.S. & Chilausky R.L. (1980). Knowledge acquisition by encoding expert rules versus computer induction from examples—A case study involving soyabean pathology. **International Journal of Man-Machine Studies**, **12**, 63-87.

Michalski, R.S. & Winston, P.H. (1985). Variable precision logic. **Artificial Intelligence**, **29**(2), 121-146 (August).

Nisbett, R.E. & Wilson, T.D.. (1977). Telling more than we can know: verbal reports on mental processes. **Psychological Review 84**, 231-259.

Pearl, J. (1986). Fusion, propagation and structuring in belief networks. **Artificial Intelligence**, **29**(3), 241-288 (September).

Rappaport (1986a). Task analysis in intelligent systems design. **WESTEX-86, IEEE 86CH2332-5.** 61-67. Anaheim (June).

Rappaport (1986b). Multiple problem spaces in the knowledge design process. **Proceedings of AAAI Workshop on Knowledge Acquisition for Knowledge-Based Systems.** 37-0-37-15. Banff (November).

Rogers, C.R. (1967). **On Becoming a Person: A Therapist's View of Psychotherapy**. London: Constable.

Shaw, M.L.G. (1980). **On Becoming a Personal Scientist**. London: Academic Press.

Shaw, M.L.G. & Gaines, B.R. (1983). A computer aid to knowledge engineering. **Proceedings of British Computer Society Conference on Expert Systems**, 263-271 (December). Cambridge.

Shaw, M.L.G. & Gaines, B.R. (1986a). Interactive elicitation of knowledge from experts. **Future Computing Systems, 1**(2), 151-190.

Shaw, M.L.G. & Gaines, B.R. (1986b). Advances in interactive knowledge engineering. Bramer, M.A., Ed. **Research and Development in Expert Systems III.** pp.111-122. Cambridge: University Press.

Swartout, W. R. (1986). Knowledge needed for expert system explanation. **Future Computing Systems, 1**(2), 99-114.

Wahl, D. (1986). An application of declarative modeling to aircraft fault isolation and diagnosis. Luker, P.A. & Adelsberger, H.H., Eds. **Intelligent Simulation Environments**, pp. 25-28 (January). La Jolla, California: Society for Computer Simulation.

Welbank, M. (1983). **A Review of Knowledge Acquisition Techniques for Expert Systems**. BTRL, Ipswich: Martlesham Consultancy Services.