

DESIGNING EXPERT SYSTEMS FOR USABILITY

BRIAN GAINES

1. Introduction

The initial success of expert system developments (Michie, 1979; Gevarter, 1983; Reitman, 1984) and the development of a number of reasonably domain-independent software support systems for the encoding and application of knowledge (Hayes-Roth, Waterman and Lenat, 1983; Waterman, 1986) has opened up the possibility of widespread usage of knowledge-based information systems. Most major new applications of information technology bring with them novel problems of user interfaces, operational procedures, training and the conceptualisation of the rôles and activities of the people and computer systems involved. Expert systems are no exception. They involve significant problems of informatics usability because:

- expert systems may be used in sensitive decision-support applications where errors are highly significant;
- expert systems are usually partially specific to the local application and require customisation;
- expert systems are usually not static and require updating according to changing circumstances and experience in use;
- expert systems usually require users to answer queries couched in specialist terms with no assurance that these terms are understood as intended;
- expert systems attempt to provide explanation facilities enabling users to understand the reasoning involved in the conclusions reached;
- expert systems often use uncertain inferences and facts which cannot be validated precisely, making it difficult to check their operations and conclusions;
- expert systems often require access to multiple forms, levels and types of inter-related data and processes, involving a complex inter-linked, multi-window interface.

In particular, the design, development and implementation of expert systems involve significant informatics usability problems in that:

- expert systems development requires knowledge acquisition from people, involving both knowledge engineers and application domain experts in specialist interactions with computing systems.

In addition, expert systems are relevant to informatics usability not only as a source of new problems but as a significant new opportunity in that:

- expert systems may be used to provide support and advice to a user of any complex information system and hence to improve the human-computer interface.

This paper considers expert systems technology as generating new modes of interaction between people and computing systems, and analyses a number of issues that arise:

- How does the nature of expert systems lead to the need for major differences in approach to systems and usability design?
- What special problems of usability arise in explanation facilities?
- What special problems of usability arise in knowledge acquisition and transfer?
- How appropriate are existing dialog design styles and guidelines?
- What additional human-computer interaction theories and guidelines are appropriate to interaction with expert systems?
- How can expert systems technology be used to improve human-computer interaction?

2. The nature of expert systems

In applying concepts of usability to expert systems it is important to distinguish those features which differentiate them from classical data processing. It is clear that expert systems use some new computational techniques, but it is perhaps less apparent that the system design approach involved is itself fundamentally changed. However, major paradigm shifts are involved (Gaines and Shaw, 1985). The classical approach in decision and control system design is the instrumentation, data collection, modelling and optimisation sequence shown in the top section of Figure 1. This approach to system design underlies the methodology 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.

However, this approach is successful only to the extent that the systems under consideration are amenable to instrumentation and modelling. Its greatest successes have been where this amenability can be achieved normatively, that is in cases where

Classical System Design	<ul style="list-style-type: none"> a) thoroughly instrument the system to be controlled or about which decisions are to be made b) use the instrumentation to gather data about the system behaviour under a variety of circumstances c) from this data build a model of the system that accounts for this behaviour d) from this model derive algorithms for decision or control that are optimal in terms of prescribed performance e) implement the algorithms as a decision or control system
Problem 1	The models available are inadequate to capture the system
Old Approach	Procrustean Design. Change the world to fit the model — normative technology
Paradigm Shift	Model Realism. Use system methodologies and information technology that enable the natural world to be modelled without distortion and destruction
Problem 2	Optimal control is over-sensitive to system uncertainties
Old Approach	Sub-optimality. Use a sub-optimal controller that is robust
Paradigm Shift	Model Uncertainty. Model the uncertainty as part of the system
Problem 3	Data is unavailable or inadequate for modelling
Old Approach	Managing. Do not automate — leave to human control
Paradigm Shift	Expert Systems. Model the person as a decision-maker or controller
Problem 4	Neither a human nor an automatic system alone is adequate
Old Approach	Ad Hoc System Design. Use a mixture of automatic and human decision/control
Paradigm Shift	Accountable Integration. Integrate automatic and human activity — make the automation accountable — “why?”

Figure 1: Paradigm shifts in system design.

the system to be controlled is itself a human artifact. For example, linear system theory has become a major tool in systems engineering not because most natural

systems are linear — they are not. The implication is in the opposite direction: that linear systems are mathematically tractable and that we design artificial systems to be linear so that we may model them readily.

The application of “a linear model with quadratic performance criterion” to natural systems is often attempted but, in general, it does not work. We have done so not because the tool was appropriate, but because it was the only one we had. However, the use of a hammer to insert screws, although partially effective, tends to distort, destroy, and generally defeat the purpose of using a screw. Similarly, the use of an inappropriate system theory to model a system may give useful, but limited, results when we have no other, but it distorts reality, destroys information and generally defeats the purpose of modelling that system.

Much of our current technology succeeds to the extent that it is normative. In agriculture we reduce the complexity of a natural ecology to a comprehensible simplicity by the use of pesticides, herbicides and chemical fertilisers. We reduce the system to one which is amenable to our modelling techniques. That simpler is not necessarily better and that re-engineering nature to impose uniformity destroys variety which is itself valuable have only been realised in recent years.

This shift in the paradigm of system design, from procrustean design to model realism, results in the first perspective shown as a response to Problem 1 in the lower section of Figure 1. Three other changes in perspective stem from this first one.

The second perspective is that which led to new techniques for dealing with uncertainty and approximate reasoning (Gupta, Kandel, Bandler and Kiszka, 1985). Optimal control theory was regarded as the peak achievement of system theory in the 1950s and 1960s. However, it proved limited in application because it demanded precision in system modelling that was impossible in practice. It was too sensitive to the nuances of system structure expressed through over-precise system definition.

The third perspective is that which is primarily associated with the stereotype of expert systems. Hayes-Roth (1984) has noted that many problems that have been felt to require human management are now amenable to expert systems. Modelling the way the expert performs the task rather than modelling the task itself is the primary characteristic of an expert system.

The fourth perspective is an important one for expert systems. They are knowledge-based systems because they make provision for explaining the decisions reached. The “why?” question has important implications for both the practical and theoretical significance of expert systems and the logics on which they are based. Expert systems represent new maturity in information technology by providing an audit trail of the data and inferences on which their recommendations are based.

Each of these paradigm shifts has implications for the usability of expert systems:

- the approach underlying *model realism* also leads to systems that conform more

closely to user's conceptual models and vocabularies;

- modelling *uncertainty* mimics human reasoning more closely but also makes the explanation and comprehension of the inference processes used more difficult;
- the *expert systems* paradigm of modelling the person, not the problem, enables advice-giving support systems to be generated for a wide variety of situations including those directly affecting usability;
- the *accountability* of expert systems through the "why?" and "how?" explanation capabilities greatly increases usability by allowing users to explore the operation of the system in comprehensive terms — however, the effective presentation of such explanations generates new usability problems in its own right.

3. The architecture of expert systems

In considering the usability of expert systems, it is important to realise that they are complex integrated systems involving a variety of tools and associated user environments, and that they also involve different categories of operator. Figure 2 shows the basic architecture of an expert 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 further enable 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 facts and inference procedures it contains;

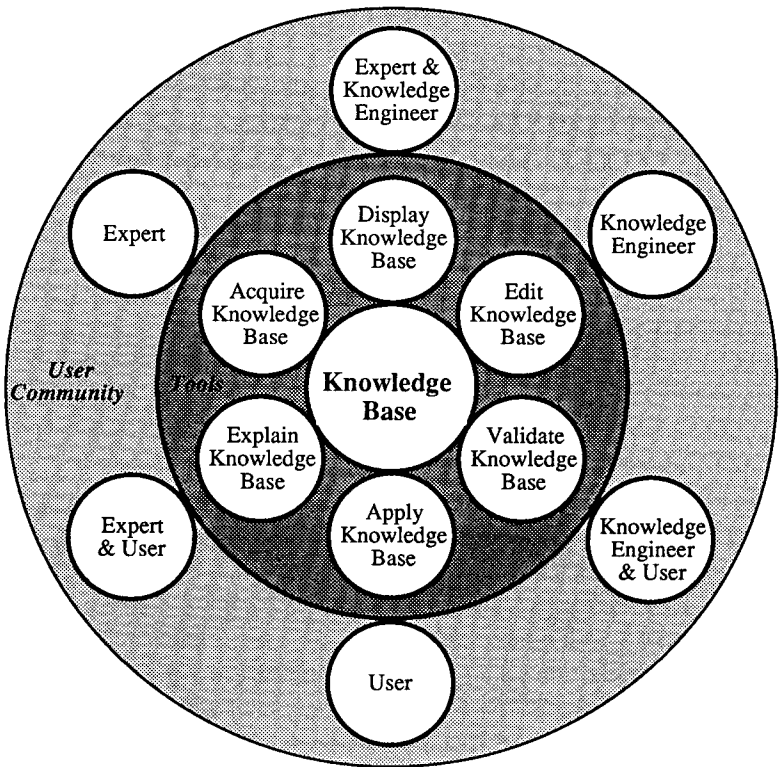


Figure 2: The architecture of an expert system.

— 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 terms of usability considerations, it is important to note that these operator rôles may be combined within a single individual, such as the 'expert and knowledge engineer', making it dangerous to assume that three self-contained user 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 2 provides a framework within which all existing systems can be embedded. 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 a user interface to the knowledge base editor which requires 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 and Winston, 1986; Pearl, 1986; Swartout, 1986; Rappaport, 1987). Most current knowledge acquisition is done by manual knowledge elicitation which places too much load on the knowledge engineer and inadequately involves the expert. However, automatic tools for rapid prototyping are becoming increasingly important (Shaw and Gaines, 1987; Gaines, 1986c; Boose and Bradshaw, 1987) and are described in a later section.

4. Usability and explanation

The significance of the explanation facilities of expert systems has already been emphasised. Explanation makes the system accountable for its recommendations. From a usability perspective it allows the user to explore the inference processes leading to the system's output, what data has been used, what inferences have been made from it, and what rules have been used in generating those inferences. Explanation was a natural feature of the production-rule backtracking approach to early expert systems. The chain of inferences from data to conclusions could easily be backtraced to provide a natural sequence of explanation. (In a later example, in

Section 7, the interaction dialog with TEIRESIAS in Figure 10 shows this backtracing used very effectively to aid debugging.)

As expert systems become more complex, however, it is unlikely that simple backtraces will be regarded as providing adequate explanations. They are, in any event, inadequate if uncertain reasoning and accumulation of evidence are involved in the inference chain — many partial inferences may contribute to the final conclusions and some ranking of their significance is required. The move towards ‘deeper’ reasoning in expert systems, involving multiple levels of inference, also raises the possibility of multiple levels of explanation. The fundamental problems of the nature of explanation in scientific reasoning are well-known in the philosophical literature (Salmon, 1984), and one should not expect easy solutions to the general requirement.

These considerations have led to a number of studies of explanation in knowledge-based systems from both psychological and technical points of view. Kidd (1985) has studied the forms of explanation requested by those asking for advice from a ‘phone-in’ radio program. Hughes (1986) has developed Lehnert’s (1977) taxonomy of forms of question to apply to the forms of explanation in expert systems. Pollack (1984) has reported a system designed to infer a user’s goals in order to give appropriate answers to badly posed questions. Swartout’s (1986) XPLAIN system demonstrates how appropriate multi-level organisation of domain knowledge may be used to generate explanations that show characteristics of dialog with human experts.

The availability of explanation in expert systems is not just a minor advance but rather a major paradigm shift towards accountable integration as shown in Figure 1. Usability considerations in explanation are as yet poorly understood, however, and major advances in both the psychology and technology of explanation may be expected over the next decade.

5. Usability and knowledge engineering

As might be expected the new technology of expert systems has also generated new problems of usability, particularly in regard to the specialist problems of its new classes of users, the knowledge engineers and experts. 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 (Hawkins, 1983) and to be able to apply this knowledge to the elicitation of expertise in specific domains.

The problems of knowledge engineering have been stated clearly:

“Knowledge acquisition is a bottleneck in the construction of expert systems. The

knowledge engineer's job is to act as a go-between to help an expert build a system. Since the knowledge engineer has far less knowledge of the domain than the expert, however, communication problems impede the process of transferring expertise into a program. The vocabulary initially used by the expert to talk about the domain with a novice is often inadequate for problem-solving; thus 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 formalise the domain concepts." (Hayes-Roth, Waterman and Lenat, 1983)

This bottleneck is the major impediment to the application of expert systems, and to the realisation of their full industrial potential. The automation of knowledge acquisition and transfer has become a major research and technological goal.

The human factors problems of expertise elicitation from a skilled person are well-known in the literature of psychology (Nisbett and Wilson, 1977; Broadbent, Fitzgerald and Broadbent, 1986). Hawkins (1983) has analysed the nature of expertise and emphasises 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 behaviour, 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 automation of skilled behaviour, and Welbank (1983) has surveyed the psychological problems of doing this. Kidd and Cooper (1985) have given a practical case history of the human interface problems of transferring knowledge to an existing expert system shell.

The main problems identified in accessing an expert's knowledge are as follows (Gaines, 1987a).

- Expertise may be fortuitous. Results obtained may be dependent on features of the situation which the expert is not controlling.
- Expertise may not be available to awareness. An expert may not be able to transmit the expertise by critiquing the performance of others because he is not able to evaluate it.

- Expertise may not be expressible in language. An expert may not be able to transmit the expertise explicitly because he is 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 the basis of a skill into skilled performance.
- Expertise expressed may be irrelevant. Much of what is learnt, particularly under random reinforcement schedules, is superstitious behaviour that neither contributes nor detracts from performance.
- Expertise expressed may be incomplete. There will usually be implicit situational dependencies that make explicit expertise inadequate for performance.
- Expertise expressed may be incorrect. Experts may make explicit statements which do not correspond to their actual behaviour and lead to incorrect performance.

In the development of knowledge engineering methodologies and 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. A trainer may be able to establish effective conditions for an apprentice to acquire expertise without necessarily understanding the skill or himself being expert in it.
- Expertise may be transmitted by evaluation. A trainer may be able to induce expertise by indicating correct or incorrect behaviour without necessarily understanding the skill in detail or himself being expert in its performance.
- Expertise may be transmitted by example. An expert may be able to transmit a skill by showing his own performance without necessarily understanding the basis of his 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 behaviour 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.

In an industrial context the application of techniques derived from clinical psychology to problems of hard systems engineering is not an obvious or probable step to take, and the early approach to knowledge engineering was to 'manage' it by inventing the profession of knowledge engineer as defined in the quotation above. It has been 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 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 labour is the appropriate solution to the knowledge engineering problem.

- 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 led 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 labour in tailoring the technology for particular applications appear to be the dominating constraint and an excessive cost.
- A move towards a labour-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 rôle 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.

These considerations of the previous section 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 (Shaw and Gaines, 1987). A brief review of such developments is given in the next section.

6. Automating knowledge acquisition in expert systems

Figure 3 shows the classical system paradigm (as shown in the top section of Figure 1) 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 computer modelling of his input-output behaviour gives better results. Textual analysis of the operator's handbook is another significant source of information about his intended behaviour.

These techniques are in fact just part of a hierarchy of knowledge transfer methodologies which defines the areas of application of the knowledge engineering techniques already in use (Gaines, 1987a). Figure 4 shows this hierarchy in relation to the basic architecture of an expert system.

- *Knowledge base*: at the centre, as before, facts and inference rules.
- *Expert system shell*: in the lower oval, the operational system for applying the knowledge base to knowledge support:
 - the inference system derives the consequences of facts about a particular situation;
 - the planning system determines how to use the inference system to satisfy the specified objectives;
 - the explanation system answers questions about the basis of the inferences made.
- *Knowledge acquisition system*: in the upper oval, the knowledge transfer processes that may be used to establish the knowledge base:
 - the knowledge generation system implements knowledge acquisition by the raw induction of models from experience without cultural support as exemplified by ATOM (Gaines, 1977) and AM (Davis and Lenat, 1982);
 - the expertise modelling system implements knowledge acquisition by mimicking an expert's behaviour as exemplified by INDUCE (Michalski and Chilausky, 1980);
 - the performance reinforcement system implements knowledge acquisition from performance feedback as exemplified by the PERCEPTRON (Rosenblatt, 1958) and STELLA (Gaines and Andreae, 1966);
 - the knowledge elicitation system implements knowledge acquisition by interviewing experts as exemplified by ETS (Boose, 1985), KITTEN (Gaines and Shaw, 1986b), MORE (Kahn, Nowlan and McDermott, 1985) and SALT (Marcus,

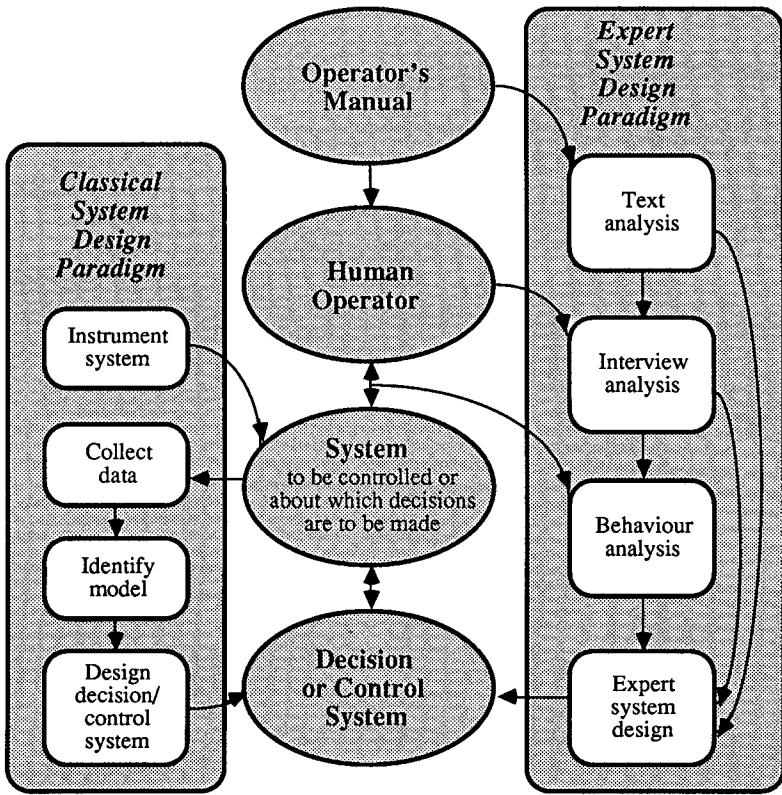


Figure 3: Expert system design paradigm.

McDermott and Wang, 1985);

- the knowledge structuring system implements knowledge acquisition by analogy as exemplified by TEIRESIAS (Davis and Lenat, 1982) and CYC (Lenat, Prakash and Shepherd, 1986);
- the basic laws system implements knowledge acquisition by building physical models as exemplified by simulation languages such as SIMULA (Nygaard and Dahl, 1981);
- the systemic principles system implements knowledge acquisition by derivation from abstract principles as exemplified by the category-theoretic language OBJ (Goguen and Meseguer, 1983).

The knowledge base and shell have been incorporated with the acquisition methodologies in Figure 4 to show the natural relationships between all three. Shell structure and knowledge acquisition research and development have so far been

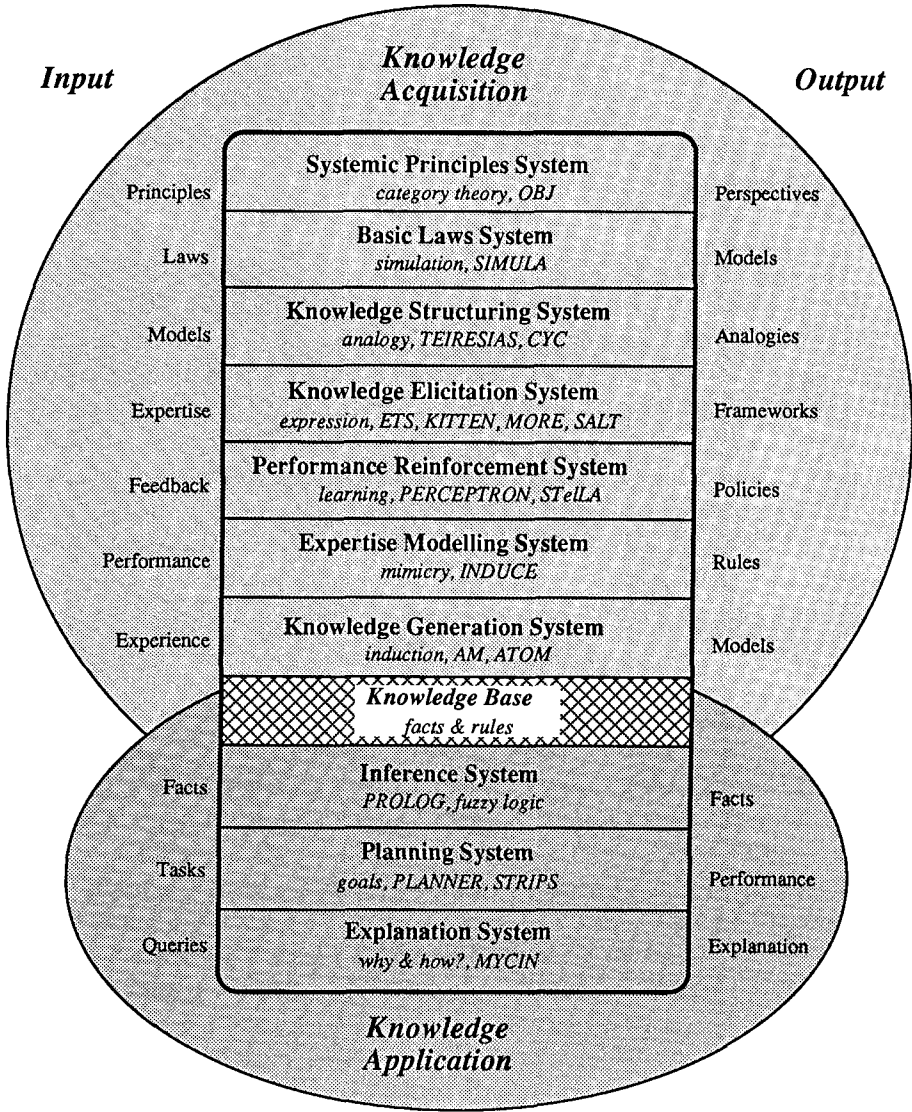


Figure 4: The hierarchy of knowledge acquisition methodologies.

treated as separate enterprises. Integrating the two is essential to future knowledge-based system applications so that the combined system provides maximal and coherent support to the expert and knowledge engineer.

The general requirements for a knowledge acquisition 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 behaviour;
- 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 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 PLANET system for repertory grid elicitation and analysis (Shaw and Gaines, 1986) is a basic knowledge acquisition system satisfying the first two requirements for domain independence and direct use. Its foundations in personal construct psychology, which itself has strong systemic and cognitive science foundations, are attractive in terms of the requirement for theoretical grounding. Boose (1985) in evaluating ETS has noted the limitations of basic repertory grid techniques in terms of requirement for a diversity of knowledge — that the methodology is better suited for analysis than for synthesis problems, for example, debugging, diagnosis, interpretation and classification rather than design and planning, and that it is difficult to apply to deep casual knowledge or strategic knowledge — and is attempting to overcome these using grid hierarchies in NeoETS (Bradshaw and Boose, 1986) and AQUINAS (Boose and Bradshaw, 1987). The TEIRESIAS extension to MYCIN is an early form of knowledge acquisition system providing debugging support for an expert system using basic analogical reasoning (Davis and Lenat, 1982).

Figure 5 shows the structure of KITTEN, an integrated knowledge acquisition system (Gaines and Shaw, 1986b; Shaw and Gaines, 1987). It consists of a knowledge base, various analytical tools for building and transforming the knowledge base, and a number of conversational tools for interacting with the knowledge base. Its operation is best understood by following sequences of activity that lead to the generation of a rule base and its loading into an expert system shell.

- A typical sequence is text input followed by text analysis through TEXAN which clusters associated words leading to a schema from which the expert can select related elements and initial constructs with which to commence grid elicitation.
- The resultant grids are analysed by ENTAIL which induces the underlying knowledge structure as production rules that can be loaded directly into an expert system shell (Gaines and Shaw, 1986c).
- An alternative route is to monitor the expert's behaviour through a verbal protocol giving information used and decisions resulting and analysing this through ATOM which induces structure from behaviour and again generates production rules (Gaines, 1976).
- These two routes can be combined. KITTEN attempts to make each stage as explicit as possible, and, in particular, to make the rule base accessible as natural textual statements rather than technical production rules.

Figure 6 shows a graphic dialog in KITTEN in which a manager is encoding the dimensions of his thinking in relation to his expertise in personnel assessment. He has entered the distinction *need supervision* — *don't need supervision* as a way of distinguishing among a triad of three subordinates randomly presented to him, and is now placing his other subordinates along this dimension by clicking on their names and dragging them to the rating scale. This natural visual approach to the expression of knowledge is very easy to use, and it is interesting to relate it back to Smith's original rationale for graphic interfaces in his development of *Pygmalion* — that it was natural to human creative thinking (Smith, 1977).

KITTEN analyses the data as it is entered and attempts to use it to continuously extend the range of dimensions of the expert's thinking in the relevant problem domain, feeding back queries based on the knowledge already entered. Figure 7 shows another graphic dialog relating to the match between two dimensions and suggesting that another subordinate be entered who might reduce this match. A similar approach is taken to matches between the subordinates themselves, prompting the elicitation of a new distinction. Such feedback tends to change the expert's focus of attention, triggering off thought processes relating to dimensions of his thinking that might otherwise be missed. It is interesting to compare the KITTEN dialog of Figure 7 with the TEIRESIAS dialog shown in the next section in Figure 11. Note how in both cases the reasoning within the system is used to focus attention on very specific

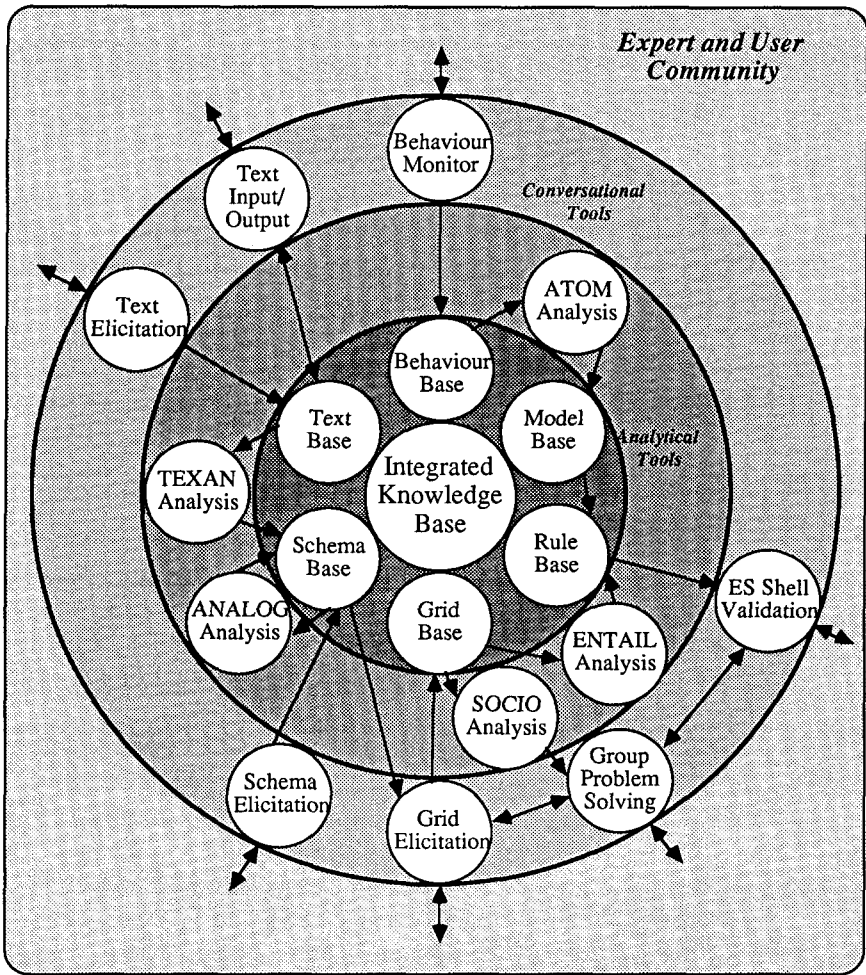


Figure 5: KITTEN – Knowledge Initiation and Transfer Tools for Experts and Novices.

issues and guide the expert in expressing just that knowledge that would improve the performance of the system. This directed guidance, while still leaving the expression of knowledge very open-ended, is the key to effective automation of knowledge engineering.

The group problem-solving component of KITTEN is particularly important because it goes beyond the stereotype of an 'expert' and 'users', and allows the system to be used to support an interactive community in their acquisition and

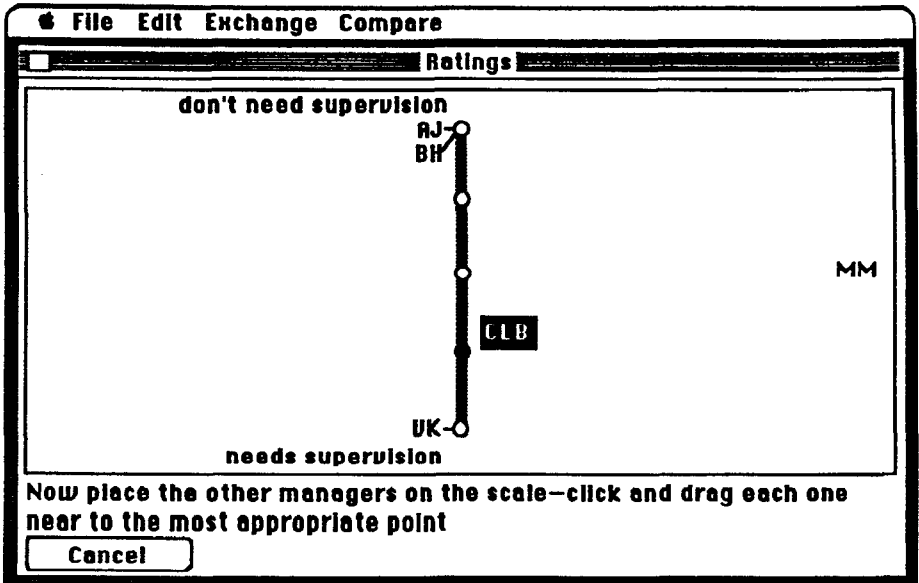


Figure 6: Graphic click and drag knowledge elicitation.

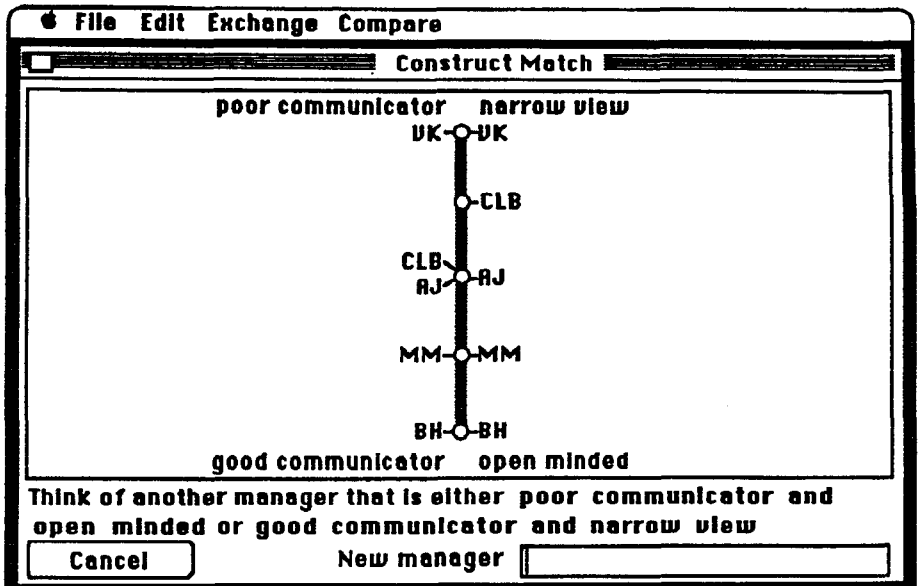


Figure 7: Feedback of matches to re-direct attention during elicitation.

transfer of knowledge and mutual understanding. The SOCIO analysis allows members of a community to explore their agreement and understanding with other members, and to make overt the knowledge network involved.

Various forms of data analysis and graphic presentation are available in KITTEN and in related systems such as ETS (Boose, 1986) and AQUINAS (Boose and Bradshaw, 1987), including the automatic generation of conceptual hierarchies and production rules with which to prime expert system shells. However, much of their success in aiding the rapid prototyping of expert systems comes from the very basic operations involved — that of eliciting the experts' own distinctions and personal vocabularies with no injection of concepts and terminologies from those outside the knowledge domain. Expert systems developed with these tools are natural to use because they operate within the same conceptual framework, using the same terminology, as normally expected by experts in the domain.

7. Dialog styles in expert systems

In the evolution of interactive systems three main styles of dialog may be distinguished (Gaines and Shaw, 1986a) as shown in Figure 8.

- *Graphic Dialog*: in which the human manipulation of objects within the world to communicate information and commands is simulated to access the activities and data structures within the computer.
- *Formal Dialog*: in which the activities and data structures within the computer are presented externally in a direct representation with the minimum syntactic sugar necessary to aid human cognition.
- *Natural Language Dialog*: in which the human use of language to communicate information and commands is simulated within the narrow context of the activities and data structures within the computer.

All three styles are found in expert systems with the variations being primarily based on the underlying technology rather than consideration of user needs.

As shown in Figure 8, the trend in the fifth generation is towards the integration of all three styles and recent expert systems reflect this. For example, GURU (Holsapple and Whinston, 1987) is a decision support shell incorporating an expert system shell that makes available a combination of formal dialog through prompt-response, menus or form-filling, coupled with natural language interaction. This makes it possible to access the same knowledge base through a variety of interactive styles that may be made to correspond to existing user experiences and preferences. Since it is designed to operate on personal computers without graphic facilities, it does not offer graphic dialog through icons and a mouse, although it would not be intrinsically difficult to incorporate this in its architecture.

Generation	Graphic Dialog <i>Representing world</i>	Formal Dialog <i>Representing computer</i>	Natural Language Dialog <i>Representing person</i>
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 Natural language output 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 Understanding DBMS domain Meta-level understanding
5 1980-87	Low-Cost PC flight simulators	Integrative	Sophisticated
	◀ Integration through INTELLECT & GURU ▶		
	◀ Integration through Macintosh ▶		
	◀ Integration through Symphony & Framework ▶		
	◀ LISP machine applications ▶		
	◀ Fifth generation objectives ▶		

Figure 8: The development of styles of dialog through generations of computers.

The best known early expert system, MYCIN (Shortliffe, 1976) was developed prior to both the Lisp machine and personal computer technologies and used a simple prompt-response dialog. It aids a clinician to make decisions about the diagnosis and treatment of microbial infections using rules of the form shown in Figure 9(i). The illustrative dialog shows MYCIN first gathering data about the patient and then using this to make inferences about the probable infections present. The first support system for knowledge engineering was also developed as part of the MYCIN project since it was found that it was difficult to trace the rules causing errors in the deductions of MYCIN if the clinician felt that something had gone wrong. TEIRESIAS (Davis and Lenat, 1982) was added as an auxiliary expert system with expertise *about* MYCIN, and the facility to explain MYCIN’s decisions, and help the clinician amend the rules when they lead to incorrect conclusions. It uses a similar rule-based approach to

RULE 50

If 1) the infection is primary-bacteremia, and
 2) the site of the culture is one of the sterile sites, and
 3) the suspected portal of entry of the organism is the
 gastro-intestinal tract,
Then there is suggestive evidence (.7) that the identity of
 the organism is bacteroides.

(i) Rule from MYCIN medical diagnosis system.**METARULE003:**

If 1) there are rules which do not mention the current
 goal in their premise
 2) there are rules which mention the current goal in
 their premise
Then it is definite that the former should be done before
 the latter.

(ii) A TEIRESIAS metarule.

Figure 9: Examples of expert system inference rules.

reasoning as does MYCIN but the rules are now *metarules* about the forms of rules and the use of rules as shown in Figure 9(ii).

The effect of the metarules on the dialog is that the clinician and MYCIN-TEIRESIAS can discuss the rules and inferences made as well as the patient, organisms and infection. Figure 10 shows a debugging sequence in which the inference made by MYCIN is noted to be incorrect and TEIRESIAS aids the clinician in correcting it. In this dialog the clinician noticed an identity was missing in those deduced by MYCIN. He chose to allow TEIRESIAS to guide the debugging, and it used the metarule of starting with the identity which appeared incorrectly. The program then reconstructed the deductive process, asking the clinician to confirm its actions at each stage, or comment on anything incorrect. It confirmed the facts about the particular case in hand and checked that the rules were correct. Before very long the issue has become sharply focused for the clinician and the system.

Many significant expert system developments in recent years have taken place on Lisp machines which provide a powerful graphic dialog environment derived from the Xerox Star design (Smith *et al.*, 1983). This has proved particularly significant in the implementation of the knowledge base display module in Figure 2 where a network representation of the conceptual hierarchies being used is very effective. This approach has been taken down-market also with expert system shells such as

INFECTION-1 is PRIMARY-BACTEREMIA
 <Item 1> E.COLI [ORGANISM-1]
 <Item 2> PSEUDOMONAS-AERUGINOSA [ORGANISM-1]
 Are all these identities correct? N
 Are there any identities missing? Y
 Which one(s)? KLEBSIELLA-PNEUMONIAE PROTEUS-NON-MIRABILIS
 Are there any identities that shouldn't have appeared? Y
 Which one(s)? PSEUDOMONAS-AERUGINOSA
 Do you want to
 1) track down the problem yourself, or
 2) allow me to guide the debugging, or
 3) ignore it for now and fix it another time?
 Type 1, 2 or 3 2
 We'll start with the PSEUDOMONAS-AERUGINOSA first. The following rules were used
 in deducing that the identity of ORGANISM-1 is pseudomonas-aeruginosa
 RULE184
 Since [1.1] the category of ORGANISM-1 is not known
 [1.2] the gram strain of ORGANISM-1 is gramneg
 [1.3] the morphology of ORGANISM-1 is rod
 [1.4] the aerobicity of ORGANISM-1 is facultative
 There is weakly suggestive evidence (.3) that the identity of ORGANISM-1 is
 pseudomonas-aeruginosa
 Is this rule correct? Y
 Should its premise have failed for this case? Y
 Which clause of the premise should have been false? [give #] 1
 No conclusions were reached concerning the category of ORGANISM-1. Should it
 have been possible to conclude a category? Y
 Which one? ENTEROBACTERIACEAE
 There were no rules successfully invoked in the attempt to deduce that the category
 of ORGANISM-1 is enterobacteriaceae.
 The following (potentially applicable) rules were NOT executed because the starred
 [**] clause failed
 RULE116
 If
 1) the identity of ORGANISM-1 is not known
 ** 2) the gram strain of ORGANISM-1 is not known
 3) the morphology of ORGANISM-1 is not known
 4) the site of CULTURE-1 is csf
 5) the infection is meningitis
 6) the age (in years) of Albert Hall is less than or equal to .17
 then there is weakly suggestive evidence (.33) that the category of ORGANISM-1 is
 enterobacteriaceae
 RULE050
 If
 1) the morphology of ORGANISM-1 is rod
 2) the gram strain of ORGANISM-1 is gramneg
 3) the aerobicity of ORGANISM-1 is facultative
 ** 4) the infection with ORGANISM-1 was acquired while the patient was
 hospitalised
 then there is suggestive evidence (.7) that the category of ORGANISM-1 is
 enterobacteriaceae
 Are all these rules correct? Y
 Should any of them have been successfully invoked? N
 Is there a rule missing? Y
 Ok, then I need a rule that would allow me to deduce that the category of ORGANISM-1 is
 enterobacteriaceae.
 Would you like to give me such a rule now? Y

Figure 10: A knowledge engineering dialog with TEIRESIAS.

Nexpert (Rappaport, 1986) which operate in the Apple Macintosh environment. Figure 11 shows some of the uses of windows in Nexpert.

- *Transcript* window keeps a log of the complete interaction, data entered, hypotheses entered, inferences made, conclusions reached, and so on.
- *Categories* window shows the conceptual structure of the rules and data.
- *Rules* window shows the rules in the knowledge base.
- *Hypotheses* window shows the hypotheses entered.
- *Data* window shows the data entered or inferred.
- *Hypothesis* window shows the current hypothesis under consideration.
- *Conclusions* window shows the conclusions reached during inferencing.
- *Prompt* window (a one-line strip at the bottom of the screen, just visible in Figure 11) shows what action would take place if the mouse were to be clicked at the current cursor position.

The prompt window is an important contribution to the usability of a complex window, icon and mouse environment where the user often finds it difficult to determine which of the many actions available will achieve his objective. As he moves the cursor around the screen using the mouse the text in the prompt window changes

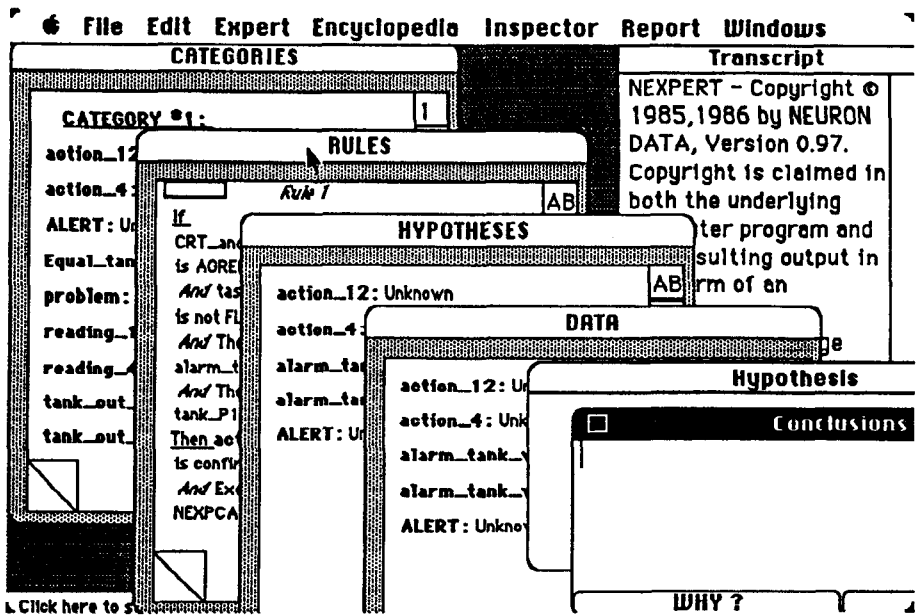


Figure 11: A modern multi-window, menu and icon environment in Nexpert.

continuously to show what action would be initiated if he clicked the mouse in that position.

Figure 12 shows a further window, the *rule editor*, which illustrates some of the care taken about human factors in Nexpert's design:

- the use of the four “head” icons at the bottom to indicate how to add a rule, modify a rule, make a copy of a rule, and delete a rule, respectively;
- the use of the “writing” icon at the bottom right to indicate how the rules may be written to disk;
- the use of the “page turning” icon at the lower left of the window to indicate how the next rule may be accessed;
- the use of the “alphabetic index” icons on the right to indicate how the rules may be accessed by the first letter of their conclusion.

Note also how the rule itself is automatically formatted for clarity in presentation. The prompt window is particularly helpful for initial users who are not familiar with the icons since moving the cursor to an icon brings up an explanation of the action involved as a prompt.

The graphic dialog capabilities used in Nexpert are important not only in their

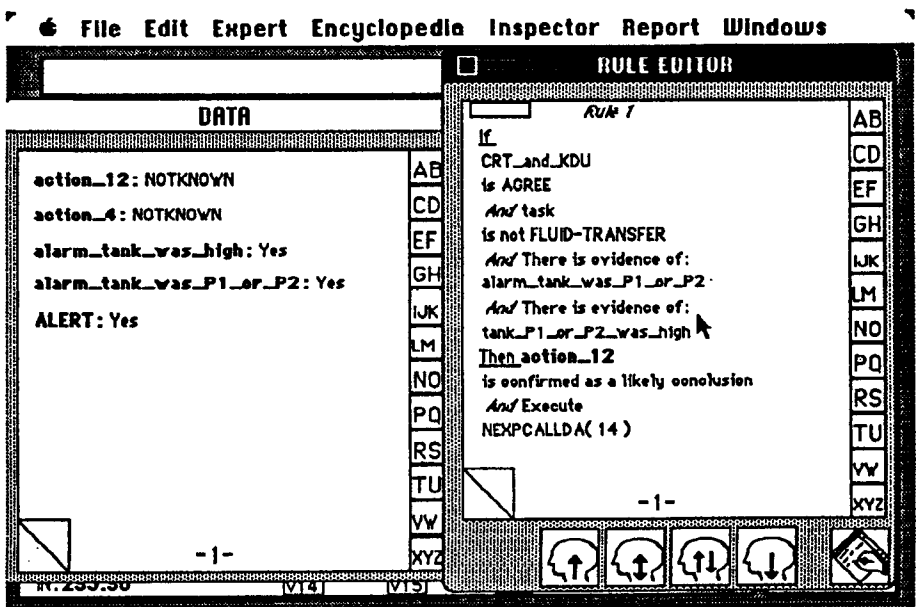


Figure 12: The rule editor dialog environment in Nexpert.

standard window, menu and icon forms, but also in their capability to allow mimic diagrams to be used in highly customised user interfaces. Figure 13 shows a mimic diagram monitoring the status of a chemical plant through data received from instruments communicating through a Macintosh serial port. The operator can see the status of the plant through the numeric indicators and also actuate the plant by clicking on the valve symbols causing them to open and close.

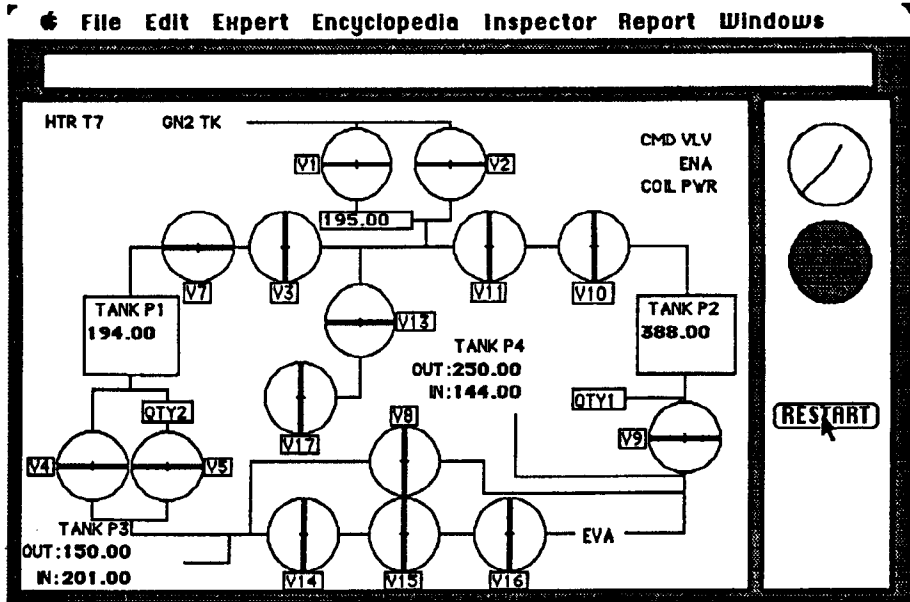


Figure 13: Dialog through a mimic diagram of gauges and semaphores in Nexpert.

Nexpert also uses the graphic capabilities to show the data and inferences in graphic form as shown in Figure 14. The knowledge engineer can navigate the knowledge base during an interaction and determine which rules have been invoked, and which rules are capable of being invoked, based on the data entered through the instruments and by the operator.

An important capability of Nexpert is that it allows a user to enter a hypothesis which it takes as a goal to be derived from inference rules and facts in the knowledge base. However, if it disproves the hypothesis, it takes the data gathered from the user during its attempt to derive the hypothesis and draws whatever conclusions it can from them. The result is that it often comes up with a hypothesis that the user regards as the “nearest” true one to that which he suggested. This is a powerful feature in an

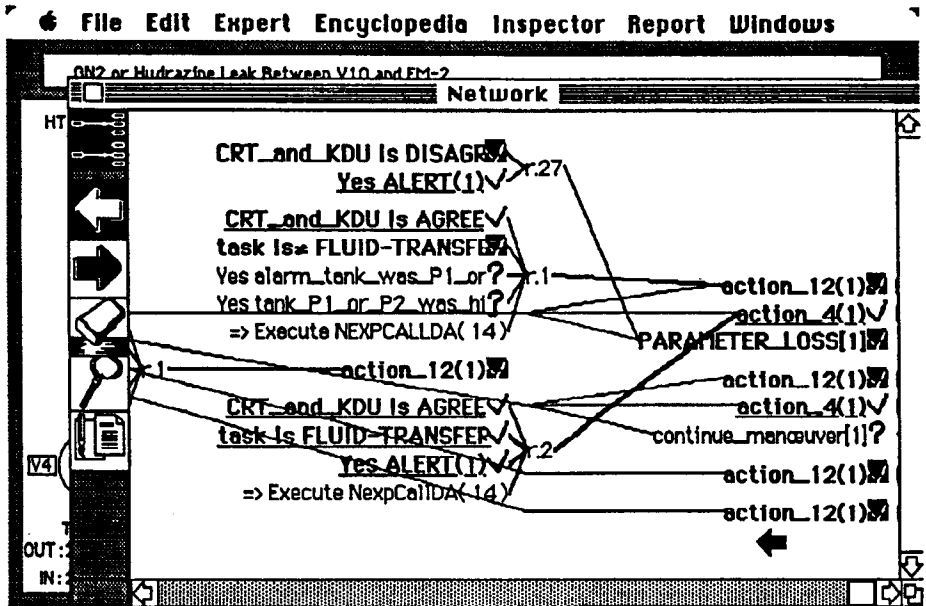


Figure 14: Graphical presentation of the relations between data, rules and inferences during a consultation.

expert system, but illustrates the complexity of user interface considerations involved at a high cognitive level.

8. Standard dialog guidelines applied to expert systems

As experience in the design and use of interactive systems has grown, guidelines have been developed for the design of effective human-computer dialog, commencing with Hansen (1971) and culminating in the very detailed review of Williges and Williges (1984). Shneiderman (1980) has collected many of these rules together, and Maguire (1982) has analysed some of them systematically, noting contradictions and proposing their resolution. It is interesting to apply such guidelines to expert systems, both to use them to design the user interface and also to see how well the guidelines apply in the context of new technology. The following sub-sections recapitulate the 30 guidelines given by Gaines and Shaw (1984) and discuss their application to expert systems.

8.1 Principles of responsibility

8.1.1 We are all responsible for computer behaviour

Remember when you treat computer-people interaction as if it were people-people interaction that the computer is behaving as it was programmed to. We, as specifiers, programmers and users, are all responsible for the behaviour of computer systems.

This guideline emphasises a particularly sensitive issue relating to expert systems — who is responsible for their advice and the adverse effects of actions resulting from errors in the system? It is surprising that accounting, medical and military use of conventional data-processing has not already led to major issues of legal responsibility for computer malfunction. Expert systems introduce a very direct link between the human professional and the information processing of a computer system and, since professionals can be held liable for the adverse consequences of their advice, it is reasonable to ask whether that liability perseveres when their advice-giving procedures are transferred to the computer.

8.1.2 Computers provide a new medium for communication

Remember when you try to understand the rôle of computers that they provide a new medium for communication that will be used in part to mimic those already existing. However, they will also change our society and modes of thinking in ways that we are not able to predict.

Books have long been used to encode expertise. However, the interaction with a book is one-way. The medium does not make provision for us to ask questions if we do not understand the book. Expert systems may be seen as the initial development of two-way, interactive books. The forms of knowledge that can be encoded and the richness of interaction possible are still very limited compared with those with the human author. However, as expert systems develop they will provide a radically new medium for human communication.

8.1.3 Computer programs encode expertise

A computer program encodes the expertise of a person to make it accessible to others. The effectiveness of dialog depends on encoding expertise in communication.

Expertise in communication can be encoded in an expert system and, hence, one major application is to provide better interfaces to existing programs.

8.1.4 Programs create the reality experienced by the users of computers

Remember that the computer is a tool for simulation and that what is simulated

becomes reality for the user. The power of the computer should be used to create worlds that are simple for the user and natural to the task.

Most expert systems currently operate through keyboard input and textual output. However, the combination of expert systems and simulation techniques to provide knowledgeable environments is extremely powerful and will be an increasing basis for significant applications.

8.1.5 The rôle expected of the system should be maintained

Tricks of dialog should not be used to give a false impression of the capabilities of the computer system. A rôle should be accurately projected that can be maintained throughout the interaction.

The rôle integrity of expert systems is particularly important if they are to be used in error-sensitive applications. Computer advice still bears the stamp of unreasonable authority. Uncertain advice should be clearly presented as dubious. If there is more than one option, alternative advice should be made available.

8.2 Principles of past experience

8.2.1 Users already have expectations about computers

Take into account the possibility that the user's expectations of the computer will affect his interpretation of any dialog with it. The dialog should be designed to minimise confusion arising from these prior expectations.

The dialog conventions of early expert systems have already become well-established so that "how" and "why" questions are expected and well-understood.

8.2.2 Choose through experience

Conversational systems should be experienced before they are talked about. Prospective users should experience interaction with a related system before specifying their requirements for their own system.

The emphasis in expert system design has been in getting the experts to act as specifier rather than the users, presumably on the assumption that users will accept the authority of experts in relation to the computer system as well as in relation to their expertise.

8.2.3 Use the vocabulary of expert and user

Listen carefully to conversations between experts and users, and design the dialog using their normal vocabulary.

The vocabulary in expert systems is very important. The expert specifies his conceptual framework and inference rules, and it is assumed that the user can

communicate and understand facts within that framework.

8.2.4 Users readily think of computer systems in the same way as they think of people

One expectation of computers is that they will behave similarly to people in their conversation and modes of operation. If we do not wish this to occur then we have to be careful that the program presents the computer as a piece of equipment and not as the simulation of a person.

It is natural to adopt an animistic approach to expert systems because they are expected to mimic the advice-generating processes of a human expert. The danger that the user will take the system outside its range of competence yet still assume it to be expert is still highly significant.

8.3 Some principles of understanding

8.3.1 Make the system easy to understand

Users will model the computer system and form new expectations based on their interaction with it. The system should be designed to induce accurate models and correct expectations.

The system needs to project to the user the scope and limitations of its knowledge base and inference processes.

8.3.2 The system should be consistent in operation

The commands should always do the same thing throughout. The information presentation should always mean the same thing throughout.

Consistency needs to apply not only to the user interface but also to the access to the knowledge base and inference system through that interface.

8.3.3 The system should be uniform in operation

The facilities which users have learned to use in one part of the package should be available to them in other parts if they might reasonably expect this.

Uniformity needs to apply to the forms of knowledge representation as well as to the user interface to those forms. Some expert systems still drop into Lisp-like data structures for certain editing operations.

8.3.4 The state of the system should be clear to the user

Computer systems can be complex and their internal state is not easy to see. Important information about the state of the system should be shown to the user in such a way that it is easily assimilated.

The state of an expert system is very complicated. The user needs to be able to review

the facts he has entered, the facts already in the knowledge base, those inferred, and the inference rules in the knowledge base.

8.3.5 The user should be shown the choices available

At any point in a formal dialog sequence the user will have a limited range of options available. There should be a facility to enable the user to find out his choices.

Queries to the user in expert systems are often open-ended yet the values tested in the rules are often highly restricted. The user should be able to determine significant replies which will actually cause different rules to be used.

8.4 Some principles of adaptation

8.4.1 The system should adapt to the user under user control

Provision should be made for various levels of user knowledge but no attempt should be made to estimate this automatically. The user should be given control over the level of help provided by the system.

Users of expert system shells vary widely in their computer literacy. Knowledge engineers, experts, frequent users and casual users, all have different support requirements that should be available under user control.

8.4.2 The system should minimise the user's workload by anticipating responses

The most likely user response should be made the default option that can be selected by a single user action.

Procedures which reduce the complexity of user decision making in expert systems are very important to the acceptance of the technology.

8.4.3 The system should cooperate with the user in validating responses

Information should be checked as it is entered and queried if it appears unlikely to be correct. The consequences of significant actions should be made clear to the user and confirmation requested before they are carried out.

Response validation can be more thorough in a well-designed expert system because the programming environment provides for the expression of complex validation procedures. Thus checking information entered should be part of the expertise embedded in the system.

8.4.4 It should be easy to escape from a dialog cleanly

At any point in a formal dialog sequence the user may wish to abort the dialog and escape any consequences of his preceding responses. There should be a

facility to enable the user to escape at will leaving the state of the system well defined.

Care is needed with the abort facility to determine which information that has been entered should be deleted. The deletion of inferences made from it is not a trivial problem, particularly if the system operates on a resource allocation rather than exhaustive inference basis.

8.5 Some principles of style

8.5.1 The style of dialog varies with the computer technology used

Some aspects of computer dialog style will be dependent on the type of person-computer interface available. Be aware of the capabilities and limitations of different interfaces.

Every form of dialog style is being used in expert systems: prompt-response; form-filling; menus and function keys; windows and mouse; and natural language. Many mixtures of these also occur and a good shell provides all styles so that it can mimic existing interactive procedures for particular user groups.

8.5.2 Be aware of the repertoire of techniques for person-computer dialog

The range of styles and techniques for dialog is continually increasing as is the experience in their use. Approach new situations with an open mind as to what techniques would be appropriate and maintain awareness of new approaches as they develop.

The availability of an expert system shell allows the dialog itself to be generated through knowledge-based procedures and hence introduces new programming styles and techniques.

8.5.3 Natural language may be used in a variety of different ways

In considering the use of natural language dialog with computer systems note that it can be used for output independently of input.

Most shells provide natural language output in prompts and explanations. This should be parametrised so that values of variables can be used within it, otherwise the external dialog may get out of step with the internal operation of the system as it is developed or modified. Writing the system for ease of modification is a significant software engineering skill in its own right.

8.5.4 Fluent language may not imply fluent understanding

Take into account when evaluating any system that apparently good conversational dialog may be generated through rules that are based on little

understanding of what is being said. There can be widely differing degrees of understanding underlying the same dialog.

Most shells do not provide a deep natural language analysis capability. Where they mimic this through some elementary keyword recognition procedures it is important that the user becomes aware of its limitations. Translating the internal representation of a natural language request back into natural language and showing it to the user is a good way to accustom the user to what is actually happening.

8.6 Some principles of presentation

8.6.1 Programs require presentation

Programs in the new medium for communication provided by computers require the same attention to techniques of presentation as have those in past media.

It is important to encourage experts to interact with, edit and validate the knowledge base of the expert system that encodes their expertise. Mediation by the knowledge engineer is an impediment to system development and should only be used for initial hand-holding to accustom the expert to the system. It is important that the system market itself to both expert and user — certainly that it places no impediments in the way of their use.

8.6.2 Know the market

The designer of a new computer program should have in mind a clear market defined by customer needs and requirements.

The top-down approach is very important to expert system design. The technology is attractive and many applications are being considered which will not prove cost-effective. The ultimate “why?” question to any expert system has to be “why should you exist?”, and it should be asked at a very early stage of development.

8.6.3 Users should be informed

At every stage in the chain from marketing through selling, training and routine use of a computer program, users of that program need to be kept informed of relevant information about it.

The problems of keeping users well-informed are exacerbated in expert systems by the complexity of the systems, their local customisation, and the need for continual local and global updating. Knowledge is complex and dynamic, and the systems reflect this.

8.6.4 Programs should be self-sufficient

Computer programs developed for a particular application should be extended to

present themselves to potential customers and to tutor their users.

Good expert system shells have the capability to present knowledge-based tutorials and provide knowledge-based help facilities. Indeed, these may be the main applications of the shell in a system that gives improved access to conventional data-processing.

8.7 Some principles of development

8.7.1 Design never ceases

Systems evolve as users gain experience and develop their requirements. The interactive facilities of the computer should be used for development also to aid the enhancement of a system in the light of experience.

Expert systems are subject to local customisation and updating, and hence the developmental tools should be fully integrated with the application environment.

8.7.2 Dialog should be logged and analysed

The computer should be used to maintain selective records of dialog and programs provided to analyse these in terms of, for example, errors broken down by user and dialog sequence.

Dialog logs are valuable with expert systems as audit trails of the advice generating process, not only to capture awkwardness in dialog but also errors in the user's understanding and the system's knowledge processing.

8.7.3 Design for a changing and uncertain future

User requirements will change, possibly unpredictably. Systems should be designed to allow for enhancements and to leave the possibility of open-ended extension.

The knowledge base in current expert systems tends to be very shallow, reflecting the surface structure of the expert's skill rather than the deeper principles behind it. Hence, the system needs to change as new knowledge is acquired.

8.7.4 Make the best use of today's technology today and tomorrow's tomorrow

Do not wait for the pace of change in computer technology to slow down. It will not. Do not assume that what is an appropriate use of today's technology will be appropriate for that of tomorrow. It may not.

Expert system technology is changing very rapidly. Shells are offering a wider variety and more powerful forms of representation and inference. User interfaces are improving. Simulation, graphics, communications and links to other programs are becoming widely available. However, we are in the early stages of a new and

immature technology and the pace of progress will increase not slacken. System designers have to make use of what is available without becoming locked in to obsolete conceptual frameworks.

9. Specific dialog guidelines for expert systems

Section 2 suggests that new informatic technologies will require new approaches to design for usability; and the analyses of explanation in Section 4, of knowledge engineering in Sections 5 and 6, and of expert system dialog styles in Section 7, have provided examples of new usability considerations in expert systems. However, Section 8 has shown that existing guidelines continue to apply. The overall situation may be captured in two *meta-principles*.

9.1 *Some meta-principles for expert systems usability*

9.1.1 Previous guidelines continue to apply

Guidelines developed for effective dialog with previous generations of interactive systems continue to apply to knowledge-based systems.

Principles of responsibility, past experience, understanding, adaptation, style, presentation and development continue to be important for new generations of knowledge-based systems. Violations of these guidelines will result in problems for the user which cannot be overcome by the use of “intelligent” processing.

Conformity with these guidelines will at least give the user a secure foundation on which to build experience of the new technologies. The interpretation of the guidelines should be widened to take into account particular features of knowledge-based systems.

9.1.2 Additional guidelines are required

Guidelines developed for effective dialog with previous generations of interactive systems do not encompass some critical features of knowledge-based systems and require specific extensions.

The critical areas where knowledge-based systems have made significant advances that affect usability are primarily in the explanation facilities and in knowledge acquisition. There are some side-effects of expert systems being part of the trend towards complex integrated systems, possibly involving multiple tasks and multiple users, that also need to be taken into account.

9.2 Principles of explanation

9.2.1 Make provision for multiple levels of explanation

Recognise the plurality of possible explanations for the conclusions reached by a knowledge-based system, and match the level of explanation to the user's requirements.

It is likely that the requirements for adequate explanation are more severe than those for valid advice, and hence this usability requirement may be a very important dynamic in the development of expert system shells.

9.2.2 Allow for ill-posed questions

User requests for explanation may well be ill-posed and provision should be made for inferring the question to be answered rather than deriving it directly from the request.

Explanation facilities may be seen as an extension of past 'help' facilities. When users requires help it is likely that they are not in a position to be clear and correct in specifying their request.

9.3 Principles of knowledge acquisition

9.3.1 Integrate knowledge acquisition with knowledge application

Make no unnecessary distinctions between the processes of applying a knowledge-based system and those of developing it.

The distinctions between expert, knowledge engineer and user are ones of rôles rather than people. Many users will play more than one rôle and will be confused by unnecessary changes in the modes of operation and data presentation.

9.4 Principles of integration

9.4.1 Generate dialog from knowledge structures

Generate all dialog directly from the same knowledge structures used for inference and do not have separate structures for questioning and explanation.

The complexity, customisation and rapidity of change of knowledge-based systems makes it very difficult to keep the main system and its support systems in step unless there are automatic links between them. Documentation, dialog and explanation should be generated from the knowledge structures rather than as separate entities. This requirement may be more severe than that for valid inference and hence should be a major design consideration for expert system shells.

9.5 Principles of rationality

9.5.1 Match the behaviour of the system to the cognitive expectations of the user

Structure dialog, inference and explanation to appear as the reasonable behaviour of an expert advisor.

The generation of requests for data through backtracking is known to appear reasonable to users. Nexpert's technique of using data elicited to test a false hypothesis to generate the 'nearest' true hypothesis also appears very reasonable to users. We need greater understanding of these criteria of rationality in order to optimise the usability of knowledge-based systems.

Many more dialog rules at greater levels of detail could be generated for informatics usability of expert systems. However, the following section suggests that it may be more productive to use knowledge-based systems to express the knowledge structure underlying effective human-computer interaction, and derive specific guidelines from general considerations.

10. The impact of expert systems on usability

The move to knowledge-based systems is not just creating new problems of usability, but also offering new solutions to old problems, including direct application to the user interface itself and indirect application to our conceptualisation of its design. The direct application of expert system techniques to improve the human-computer interface is an attractive possibility, for example, in replacing simplistic *help* facilities with a more flexible expert advisor. Such possibilities have been studied in the literature under the heading of *intelligent user interfaces*, and Rissland (1984) gives a number of knowledge requirements for intelligent interfaces:

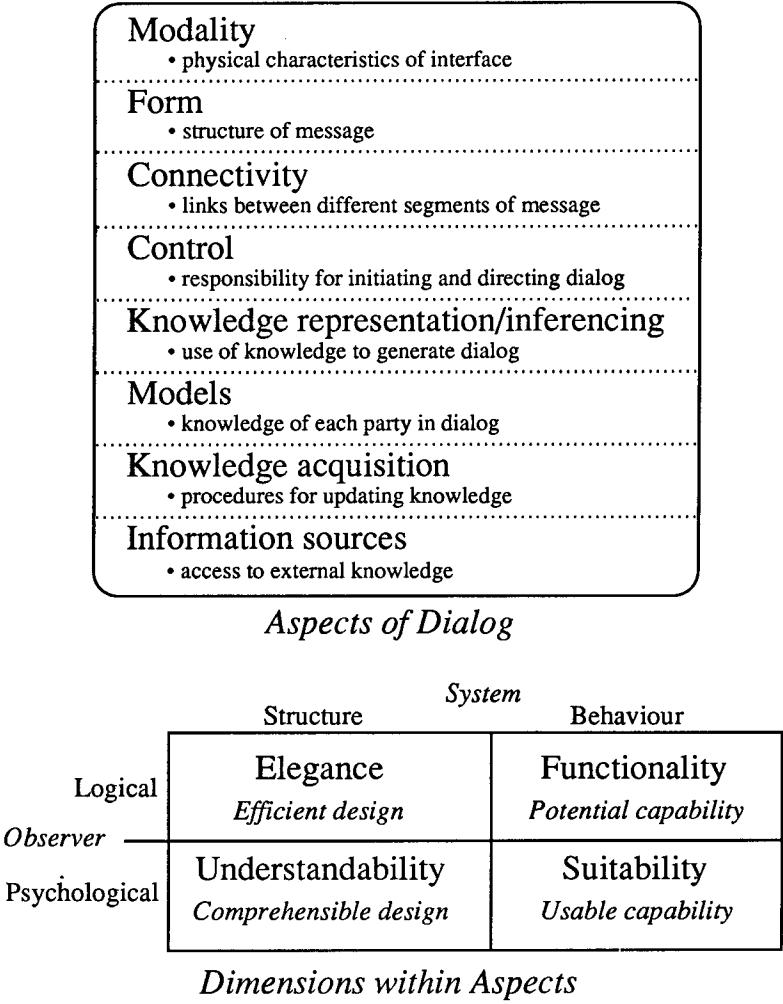
- knowledge of the user — for example, preferred style or level of expertise;
- knowledge of the user's tasks — for example, context or ultimate use of results;
- knowledge of the tools available and being used — for example, protocols for invocation and use of resources needed;
- knowledge of the domain of the user's task — for example, epistemological knowledge or measures of importance;
- knowledge of interaction modalities — for example, how to communicate graphically or by speech;
- knowledge of how to interact — for example, when to intervene or look over the user's shoulder;
- evaluation knowledge — for example, how to judge effectiveness or act on gripes.

There are few examples of practical applications of intelligent user interfaces as yet. Totterdell and Cooper (1986) have reported a study of an adaptive front-end to an electronic mail system. Chalfen (1986) has developed an expert shell for easing user access to integrated heterogeneous software. This is particularly important in enabling third generation simulation software already existing to be brought into a fourth generation interactive environment through the use of fifth generation knowledge-based techniques. Kitto and Boose (1987) have described an intelligent user interface to a knowledge acquisition system which guides the expert through the complexities of the acquisition process. This is significant in demonstrating that the new problems created by knowledge-based systems may also be alleviated using knowledge-based techniques.

The growing complexity of guidelines for informatics usability suggests that the use of knowledge-based systems for the design and evaluation of human-computer interaction may be attractive. Wilson, Barnard and MacLean (1986) have described an expert system development that makes user interface guidelines available as a decision support system for system design. Gaines (1987b) reports a systemic analysis of the knowledge structures underlying a human factors evaluation methodology for complex multi-modal, multi-user, multi-task systems developed by Edwards and Mason (1987).

Figure 15 is a synopsis of the Edwards and Mason evaluation methodology. They analyse an intelligent dialog system in terms of the eight aspects shown at the top of Figure 15, and then evaluate it in terms of the four dimensions shown at the bottom of the figure: *elegance*, capturing the notion of efficient design; *understandability*, capturing the notion of comprehensible design; *functionality*, capturing the notion of potential capability; and *suitability*, capturing the notion of usable capability. The knowledge structure of the basic distinctions underlying this evaluation methodology arise from the basic distinctions of structure and behaviour, and the evaluative dimensions arise from the basic distinctions of actuality and agency.

The development of knowledge structures underlying human-computer interaction will enable the empirical rules for effective dialog design to be subsumed into deeper principles applying at each level of the hierarchy of analysis of intelligent interactive systems. These structures are also the knowledge engineering foundations for knowledge-based advisory systems on complex human-computer system design and evaluation. Thus, as usual, progress in informatics is recursive with the next generation of knowledge-based systems providing both a range of new problems and also a range of techniques to cope with them.



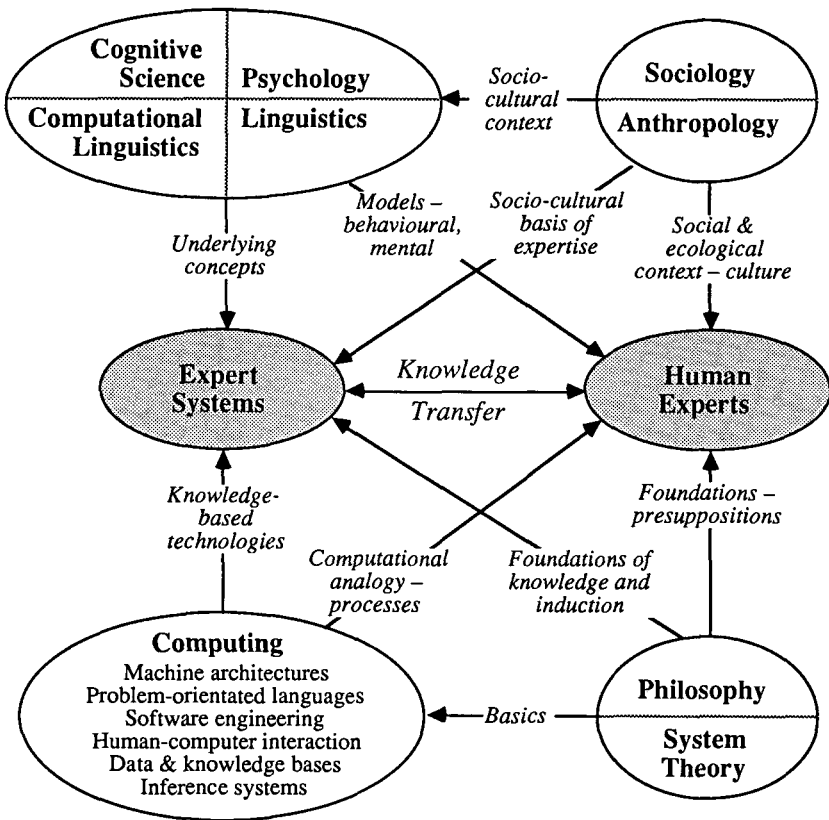


Figure 16: Relationships between the many different disciplines contributing to the usability of expert systems.

the relevance of existing human-computer interface guidelines and the new problems that arise from these systems. In conclusion it is interesting to note the very wide range of disciplines necessary to support the scientific development of usability considerations for expert systems. Figure 16 shows the wide range of relationships and contributions necessary:

- The fundamental problem is that of *knowledge transfer* between computer-based expert systems and human experts of various kinds, including users and knowledge engineers as well as domain experts.
- *Computing science* provides the required technologies with special-purpose machines, languages, software engineering techniques, human-computer

interaction guidelines, data and knowledge base systems, and inferencing systems for logical and uncertain reasoning.

- *Philosophy* and *system theory* provide the foundations for understanding the nature of knowledge, its representation, acquisition and processing.
- *Psychology* and *linguistics*, and their computer-influenced developments in *cognitive science* and *computational linguistics*, provide models of the person, thought processes, conversational interaction, and so on.
- *Sociology* and *anthropology* provide models of the formation of expertise and its rôle in society, and of the cultural processes involved in human knowledge acquisition which have to be emulated in computer-based knowledge acquisition.

What Figure 16 suggests is that we should expect no easy answers to the problems of informatics usability in the increasing development and utilisation of knowledge-based systems. They are not just a simple extension of existing informatics technology. Our existing guidelines apply but they are inadequate and need major extensions. As we complete the fifth generation era of information technology development with its emphasis on the hardware and software architectures of knowledge-based systems (Moto-Oka, 1982; Gaines, 1984), proposals have already been made for sixth generation programs of research (STA, 1985; Gaines, 1986a, 1986b). It is interesting to note that these call for the development of knowledge science based on inter-disciplinary collaboration between neurology, psychology, linguistics, logic and computing science. Such inter-disciplinary interchange is necessary to the deep understanding of the usability of knowledge-based informatics systems.

Acknowledgements

Financial assistance for this work has been made available by the National Sciences and Engineering Research Council of Canada. I am grateful to my colleagues John Boose, Jeff Bradshaw, Alain Rappaport and Mildred Shaw for many discussions and access to material used in this paper.