CHAPTER 3

KNOWLEDGE ACQUISITION SYSTEMS

BRIAN R. GAINES

1 KNOWLEDGE-BASED SYSTEMS

Knowledge acquisition is recognized as a major bottleneck in the development of knowledge-based systems. The first generation of knowledge acquisition techniques in the 1970s was based on human interviewing. The second generation in the 1980s has been based on computer-aided application of specific techniques such as repertory grids, behavior modeling, and text analysis. Current third-generation research is targeting tools for the 1990s which will provide highly integrated knowledge acquisition environments offering a wide range of complementary tools and techniques. This chapter surveys the state of the art in knowledge acquisition, providing a framework for the wide range of techniques in use, describing the major ones in detail, and illustrating them. It aims to give a solid foundation for tool developers and an indication of appropriate techniques for system developers.

This chapter focuses on the *acquisition* of knowledge for knowledge-based systems. It is primarily concerned with the *transfer* of knowledge from existing sources, such as experts and books, rather than the *learning* of knowledge from experience. *Machine learning* is an important aspect of knowledge-based systems, but it has a major literature of its own (Michalski and Carbonell, 1983; Michalski, Carbonell, and Mitchell, 1986) which is in large part distinct from issues of knowledge and expertise transfer from existing sources. In a broad sense, learning and transfer processes are both part of "knowledge acquisition" for knowledge-based systems and have many issues in common, particularly those of knowledge representation and inductive modeling. However, most practical knowledge-based systems currently acquire

their knowledge base through the transfer of existent knowledge rather than learning from experience, and a major literature has developed about such transfer, its problems, and tools and techniques to overcome them (Gaines and Boose, 1988; Boose and Gaines, 1988). This literature is surveyed in this chapter.

One caveat should be added to the above remarks: The notion of *existent* knowledge is rather more complex than might appear at first. Knowledge transfer is rarely a straightforward problem of obtaining knowledge in some form and encoding it for the computer. For example, experts may be able to *perform* complex tasks and yet have major problems in *understanding* and *communicating* how they do this (Bainbridge, 1986; Gaines, 1987b). Understanding and communication are skills quite distinct from performance, and individuals may have widely differing capabilities in each of these three. The practitioner, the scientist, and the teacher are three distinct roles that are developed in differing degrees in different individuals. In knowledge acquisition it may be necessary to build a framework for the understanding and communication of expertise which did not previously exist. It is these types of issues that make knowledge acquisition for knowledge-based systems a complex and fascinating domain, of great practical significance, interacting with major research issues relating to human cognition, knowledge transfer processes in society, and computational problems of supporting and automating such processes.

Figure 3-1 shows the chain of information flow between expert performance of a task and client performance of the same task supported by an expert system.

The person in the role of an expert is shown as having three associated roles, or internal processes: performing the task; understanding the basis of the performance; and communicating the basis of the performance, termed the *practitioner, scientist,* and *teacher*, respectively, in the preceding paragraphs.

The person in the role of a knowledge engineer is shown as having three similar roles: communicating about the basis of the performance with the expert; understanding the basis of the performance at a level sufficient to specify it for a computer; and programming this specification in a form suitable for communication to an expert system on a computer.

The computer in the role of an expert system is shown as having three major internal processes: communicating about the basis of the performance with the knowledge engineer; "understanding" the basis of the performance in terms of its knowledge representation and inference processes at a level sufficient to generate advice for a client; and advising on specific task performance through communication with a client.

The person in the role of a client is shown as having three roles similar to those of the expert: communicating about the basis of specific performance with the expert system; understanding this at a level sufficient to carry out the performance; and performing the task based on the advice of the expert system.

The shaded arrow between the communication roles in the expert and client indicates the advisory role of the expert that is being emulated in part through the expert system. The actual flow of information involved in this ad-

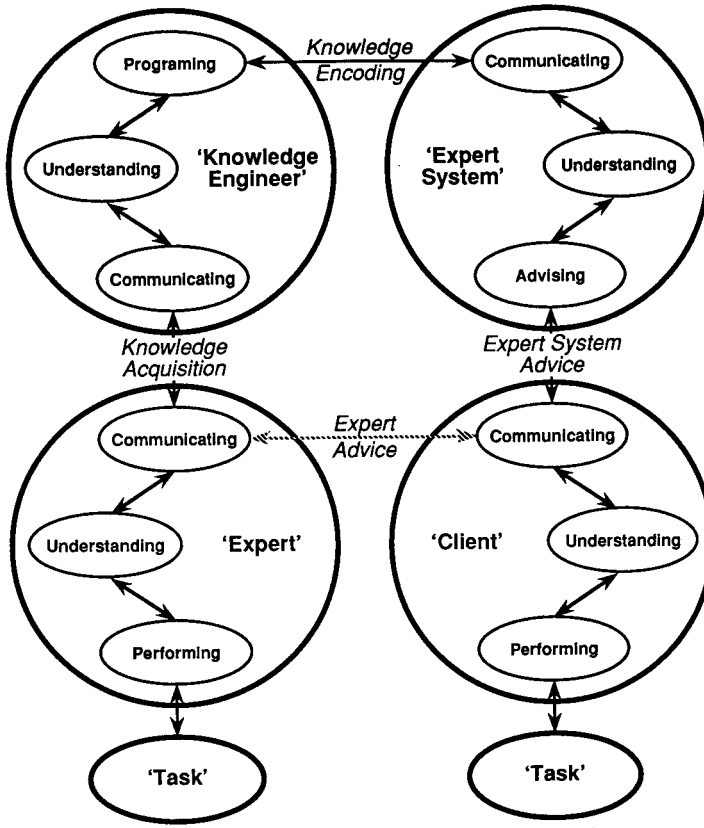

FIGURE 3-1
Information flow between expert performance and client performance.

vice consists of knowledge acquisition by the knowledge engineer from the expert; knowledge encoding by the knowledge engineer into the expert system; and then expert system advice from the expert system to the client.

The problem of expertise elicitation from a skilled person is well known in the literature of psychology (Nisbett and 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 experts, not only because they cannot express it, but also because they may not be aware of its significance to their activity. Bainbridge (1979, 1986) has reviewed the difficulties of verbal debriefing and notes that there is no necessary correlation between verbal re-

ports and mental behavior and that many psychologists feel strongly that verbal data are useless.

These are the main problems identified in accessing an expert's knowledge (Gaines, 1987b):

- "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 she is unable to express it.
- Expertise may not be understandable when it is 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 it is expressed in language. An apprentice may not be able to convert verbal comprehension of the basis of a skill to skilled performance.
- Expertise expressed may be irrelevant. Much of what is learned, particularly under random reinforcement schedules, is superstitious behavior that neither contributes to 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 behavior and lead to incorrect performance.

Thus, the many processes in the long chain of information flow shown in Fig. 3-1 are each a potential source of weakness in the transfer of expert performance from the expert to the client through a knowledge engineer and expert system. However, the fundamental issues underlying knowledge acquisition should not distract attention from the practical objectives of knowledge acquisition research: to develop effective knowledge-based systems. The concept of a knowledge-based system can be put in operational terms that do not involve any grand dreams of artificial intelligence or major advances in cognitive science.

- Computing technology is still far from the general intelligence and commonsense of people.
- However, in well-defined domains it is possible to build models of likely transactions and use them to aid and advise people.
- If these models are *explicit* (not embedded in code) and *high-level* (understandable in noncomputer terms), then it is reasonable to call the data structures involved *knowledge*.

- The advantages of such explicit, high-level knowledge-based systems are that they can be more easily *developed by end-user communities* and that the knowledge structures can be used in *explanations* such that the systems appear reasonably *intelligent.*

Put in these terms, the development of knowledge-based systems may be seen as part of the continuing evolution of data processing toward increased end-user understanding and control. This chapter encompasses both the fundamental and the practical aspects of this evolution.

2 KNOWLEDGE-BASED SYSTEMS IN COMPUTING

In the early days of systems, the nature of these systems appeared reasonably well defined. They were computer-based decision support systems that emulated the performance of a human expert based on knowledge obtained from that expert. However, as a variety of systems were developed and applied, some aspects of this definition became less clear.

- Does it matter if the system emulates an expert, if it performs well?
- Does it matter that the expertise is obtained from experts rather than other sources?
- Does it make sense to combine other decision support technologies, such as computation and simulation, and how do they interact effectively?
- Are expert systems intrinsically in a decision support role, or can they be used for direct control, for information retrieval, and so on?

What has occurred is an increasing integration of expert systems with other computational approaches, techniques, and methodologies, and a resultant blurring of distinctions.

It is useful to gain a perspective on this evolution of knowledge-based systems and their integration with conventional computing technology, by taking the viewpoint that computers were developed to support human knowledge processes, and all computing applications may be viewed in this light. What, then, differentiates expert and knowledge-based systems? Figure 3-2 classifies human knowledge processes and lists the relevant techniques and supporting systems, precomputing and postcomputing (Gaines, 1988c). The basic level is that of human *experience* as the generative substratum for knowledge processes. This has two major refinements, corresponding in engineering terms to *modeling,* allowing *anticipation* of potential future experience, and *control,* allowing the world being experienced to be changed through *interaction* with it.

In the survival of the species, it is important that experience and the lessons learned from it be shared. Others may be able to learn from experiences more effectively than those who have them. Individuals are finite and expendable, but their experience is part of the knowledge economy of the species. Hence the next level is *communication* of experience. This has two major refinements: accurate *description,* allowing others to model the experience, and

Human Knowledge Processes	*Relevant Techniques*	*Supporting Systems*	
		Pre-Computing	*Post-Computing*
Experience	Perception, Memory	Instruments, Data Recording	Instrumentation, Data Storage
Anticipation	Cognition, Modeling	Modeling Techniques, Statistics	Data Analysis
Interaction	Action, Control	Tools, Mechanisms	Simulation, Robotics
Communication	Mimicry, Rhetoric	Pamphlets, Novels, Drama, Movies	Games
Description	Representation, Language	Representational Painting & Writing, Phonograph, Television	Word Processing, Graphics, Information Retrieval
Discourse	Argument, Communicative Action	Books, Journals, Journalism, Telephone, Education System	Electronic Mail, Hypermedia, CAL, Natural Language
Reasoning	Induction, Abduction, Analogy	Jurisprudence Systems, Uncodified 'Commonsense'	Semantic Nets, Conceptual Graphs, 'Advice Systems'
Deduction	Mathematics, Set Theory, Arithmetic	Calculators, Diagraming Techniques	Number Crunchers, Relational Databases, 'Expert Systems'
Logic	Propositional & Predicate Calculi, Modal Logics	Proof Techniques, Sequent Calculi	Algebraic Manipulators, Theorem Provers
Category Theory	Universal Algebras, Topology, Homology	Metatheorems, Proof Techniques	Category-Theoretic Programing Languages

FIGURE 3-2
Human knowledge processes, relevant techniques, and supporting systems.

discourse, persuading others to adopt one's own model. Discourse allows for indirect action upon the world by persuading others to act in one's place. It involves the use of communication intentionally to affect the behavior of others and, in particular, to communicate not just experience but *understanding.*

Again, in the survival of the species, it is important that experience and the lessons learned from it be detached from the discourse processes of particular individuals, with whom interaction may not be widely available. Hence the next level is that of *reasoning,* concerned with the creation of knowledge from experience and its application to the generation of experience, i.e., determination of the meaning, significance, consequences, actions, and princi-

ples based on experience. Three major refinements are shown: the development of models and their consequences through *deduction* based on arithmetic and mathematics, formal reasoning based on *logic*, and metatheoretic foundations of mathematics and logic through *category theory*.

The techniques and support systems underlying these knowledge processes are shown in three columns in Fig. 3-2:

1 Experience
   - **a** Relevant techniques: human interaction with worlds through perception, memory, cognition, and action
   - **b** Supporting technology: instruments for enhancing perception; techniques for enhancing modeling; and tools for enhancing action
   - **c** Computing technology: the support of natural and artificial experience, anticipation, and interaction through instrumentation, data analysis, simulation, and robotics

2 Communication
   - **a** Relevant techniques: interpersonal interaction through mimicry and rhetoric; representation of experience through media and description of experience through language; mutual understanding of experience through argument
   - **b** Supporting technology: literary forms for presenting experience; painting, writing, phonograph, and television as media for capturing experience; books, journals, and the education system supporting mutual understanding
   - **c** Computing technology: games giving access to stylized experience; word processing and graphics as media representing and presenting experience; electronic mail, hypermedia, and computer-aided learning supporting mutual understanding

3 Reasoning
   - **a** Relevant techniques: inductive and abductive inference and analogical pattern recognition; set theory, arithmetic, logical calculuses, topology, and homology
   - **b** Supporting technology: surprisingly little—jurisprudence through the legal system to support justice in socially important practical reasoning, but primarily left to uncodified "common sense"; calculators, diagraming, and proof techniques at the formal level
   - **c** Computing technology: knowledge representation and inference through semantic nets and conceptual graphs; numeric computing, relational databases, and expert systems; algebraic systems and theorem provers; category-theoretic programming languages

Looking down the right-hand column of Fig. 3-2, we see the full spectrum of computer-based techniques supporting human knowledge processes. Note how the development of computing has straddled the central region of discourse and practical reasoning. Computing commenced with number

crunchers, moved into simulation and instrumentation, and then went into databases, graphics, and text processing. The support for the explanatory processes of discourse and the argument forms of practical reasoning is still very weak, even though these underlie our colloquial use of the term *expertise*. Current expert systems largely support formal reasoning and may be seen as a natural extension of database technology to allow retrieval of logically derived fields.

This weakness in the emulation of expert knowledge processes is clear in the knowledge acquisition literature where it appears as a gap between the knowledge elicited from experts and its accurate encoding for an expert system shell (Rappaport, Gaines, and Boose, 1988). The gap is being filed by research on knowledge representation schema better suited to discourse and practical reasoning, but so far these have had little impact on commercially available knowledge-based systems. Expert systems technology has developed primarily at the level of formal reasoning which is a very narrow subdomain of the human reasoning processes.

The diversity of human knowledge processes cannot be overemphasized—indeed, *anarchy* might be a better term than *diversity* (Feyerabend, 1975). "Knowledge" in the AI literature has become associated with formal structures from which inferences can be made. However, much human knowledge is not well structured for formal representation and computer processing (Gaines and Vickers, 1988). The philosophical dictionary definition of knowledge as "more than opinion, less than truth" (Runes, 1972) encapsulates what can be said. Classical data processing methods take it for granted that the data can be assumed to be "true" and base their inferences on this. In knowledge-based systems, we have data that is less than truth but more than opinion because it is "expert opinion." The implications of these notions of untruth (inconsistency, contradiction, change) and expertise (modeling, cognitive authority, reflective equilibrium) are critical to the design of knowledge acquisition systems.

The need for integration of computing technologies is very clear in Fig. 3-2. Practical knowledge support systems must be capable of providing combinations of graphics, text processing, hypermedia, number crunching, databases, communications, theorem proving, and semantic nets and expert system technologies. These are not highly separated functions but rather multiple aspects of human knowledge processing. The questions raised at the beginning of this section are also put into perspective. As part of the total infrastructure of information technology, expert systems provide a deductive reasoning capability that can be used in a variety of ways regardless of its source or relation to human expertise. However, the use of this reasoning capability in conjunction with overt, high-level data structures, or "knowledge," is a significant advance in the usability of information technology, and this does, in part, derive from the emulation of expert sources. However, once the concept and significance of overt, high-level knowledge become clear, it is less important how it is derived than how it is used.

The next section addresses the sources of expert knowledge and the role of the expert in society, giving a framework for the application of information technology and knowledge-based systems as part of the overall knowledge processes of society.

3 KNOWLEDGE ACQUISITION

The preceding section has suggested that knowledge-based systems should be regarded as an integral part of information technology and that the technology itself should be regarded as an integral part of human social processes. This section introduces a model of expert behavior and expertise which gives a practical framework for knowledge acquisition and itself suggests that acquisition should be regarded as an integral part of an expert system's performance.

The distinction between knowledge acquisition and performance systems is an invidious one. It can be taken to imply that expertise is a static collection of skills and knowledge, capable of being elicited and transferred to an expert system that will then be able to reproduce the expertise. This static model is an extremely dubious one in systems design, leading to systems that date rapidly, interfere with the growth of expertise, and have poor reliability and high maintenance costs. The following explains how these conditions arise and approaches for avoiding them.

Hawkins (1983) has abstracted from industrial experience in developing mineral exploration expert systems and proposed a model of human expertise relevant to expert systems. Figure 3-3 summarizes the essential features of this model:

- The expert first elicits data about the problem from the client.
- She or he develops a minimal model that accounts for the data provided.
- The expert generates advice based on the model and feeds this to the client.
- The client may accept the advice or query it, and possibly, the model.
- The queries lead to further data elicitation and repetition of the elicitation-modeling-advice-query cycle.

Thus, in Hawkins' model, the client plays an active role in further developing the model by providing more data until he or she is satisfied with the model and consequent advice. Expert advice giving and taking is part of a cycle of negotiation around a process of model formation.

The particular importance of this model of expert performance lies in the emphasis it places on the embedded knowledge acquisition process. The expert is managing a data collection, modeling, and decision cycle within a consultation. As Fig. 3-3 shows, this implies that the expert is able to enhance her of his collection of case histories, models, and advisory strategies through the consultation. Skilled performance is not the static utilization of expert knowledge and skills in Hawkins' model—it is, instead, a microcosm, and an essential part of, the wider processes of knowledge acquisition which are intrinsic to the role of an expert.

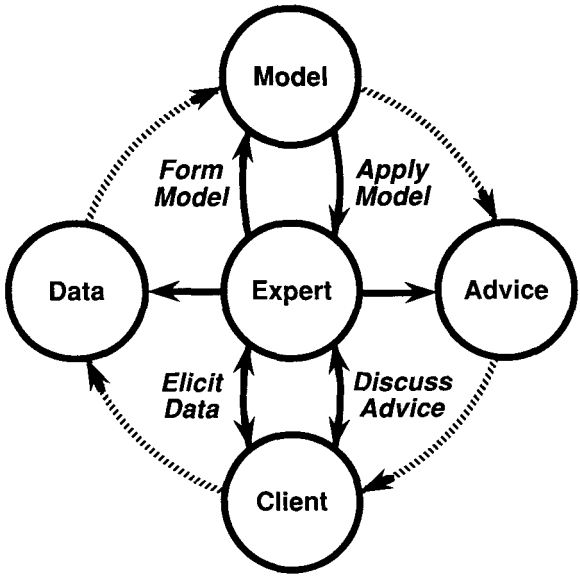

FIGURE 3-3
Negotiation cycle in expert-client interaction.

Access to the learning experiences of attempting to solve the problems of a client community is one major source of knowledge in the development of individual expertise. The client community provides access to practical experience, a variety of problems, particularly novel ones that go beyond existing expertise, and it also manages the growth of expertise through systems of reward, criticism, and access to resources. However, learning from experience is a slow and error-prone process, and socially significant areas of expertise become associated with professional communities that attempt to expedite learning and reduce errors through the sharing of experience. Figure 3-4 extends the model of Fig. 3-3 to include these other processes, such as apprenticeship, instruction, training, education, workshops, conferences, journals, and books. Professional communities usually play a major role in refining and directing the client community's reward, criticism, and resource allocation systems.

The role of the professional community can be seen as an indirect enhancement of the basic processes of knowledge acquisition in the negotiation cycle of expert performance. The individual can acquire the data of others through access to reported case histories and enhance his or her repertoire of models through the published theories developed by others to account for their data and the data of many others. The individual can enhance her or his repertoires of advice through the published advisory strategies developed by others and evaluated by them, or yet others, against past case histories, theoretical foundations, and experience in use.

This model of individual expertise as intrinsically a knowledge acquisition process at multiple levels intrinsically embedded in, part of, and sup-

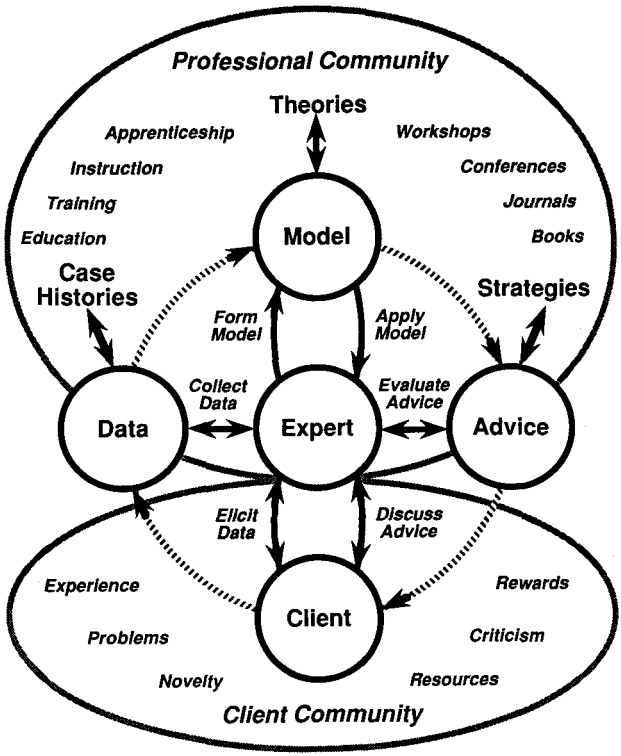

FIGURE 3-4
Extended knowledge acquisition process in expert-community interaction.

ported by essential social processes is consistent with Stich and Nisbett's (1984) view of the social role of experts as managers of the *reflective equilibrium* of the inductive process of knowledge generation in society. This model is also consistent with the emphasis in professional development books concerned with good practice on the importance of balancing experts' authoritarian roles as knowledge repositories and appliers with their entrepreneurial roles as knowledge acquirers (Schön, 1983).

Sociologists have noted the very strong positive feedback processes in the dynamics of knowledge acquisition in the scientific community (Hagstrom, 1965). Merton (1973) coined the term *Matthew effect* for those features of the reward system in research biased toward allocating greater credit for the same discovery to those with an established reputation. The qualitative effect of such positive feedback processes is to amplify random differences within a population, creating strong distinctions which have no other basis than the feedback process itself. That is, even if all scientists were created equal in their capabilities, some would become experts relative to others because ran-

dom differences in early performance generated credit that affected resource allocation and gave them greater opportunities to acquire knowledge. The strength of this effect can be demonstrated by studies of the behavior of "experts" simulated by learning automata in a competitive learning environment with access to resources related to demonstrated expertise (Gaines, 1988b).

In many areas of expertise where expert systems are expected to play a major role, these positive feedback processes are exceptionally strong. In medicine, e.g., the key learning resource is access to medical problems, and the "owner" of such a problem has a strong personal interest in allowing only someone of very good reputation to handle it. The system, including considerations of legal liability, strongly funnels problems to those who are regarded as experts. It is, however, precisely from these problems that new knowledge is generated. Hence direct experience of case history data is made available primarily to those with established reputations for having capabilities to deal with it. Similar considerations apply to the award of scholarships, invitations to scientific congresses, and so on (Blume, 1974). They also apply not only to individuals but also to social units, such as a company subject to government procurement procedures that are heavily biased to contractors with prior experience and with whom the government agency has prior experience.

The introduction of knowledge-based systems in the processes shown in Fig. 3-4 may be seen as providing technological support to the expert role in the center of the figure. It corresponds to the change from expert advice to expert systems advice shown in Fig. 3-1. The system developer operates in the same environment, and Fig. 3-4 may be restructured to show the sources available to the developer of a knowledge-based system. Figure 3-5 illustrates the knowledge acquisition process with the system developer drawing on three major sources: knowledge encoded in *relevant media,* such as books, journals, and videotapes; knowledge available by observing and modeling the *relevant domain;* and knowledge available from discourse with, and observation of, the *relevant community*. It is this last source that is most associated with expert systems as such, although all three sources play significant roles in the development of any practical system.

Note that the *relevant community* has been split into requirers, experts, and client. Again, domain experts are the expected source of knowledge for an expert system, but detailed requirements specifications and involvement of end-users play as significant a role as they do in other computing system design. It is useful to think of those with these different roles as all being experts in some different aspect of the system functioning, since the knowledge acquisition techniques used with domain experts are often well suited to establishing requirements specifications and end-user perspectives.

The annotation of the *knowledge base* in Fig. 3-5 illustrates the range of technical features which an adequate knowledge-based system shell may be expected to support:

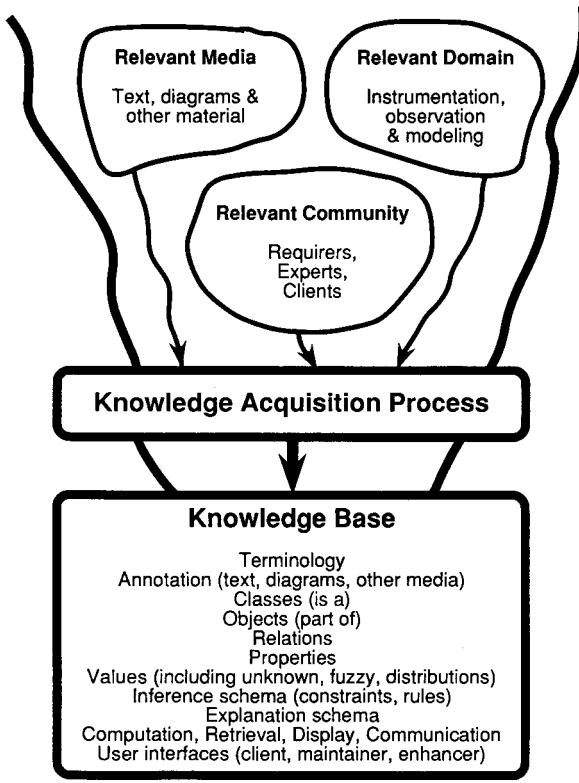

FIGURE 3-5
Knowledge acquisition process for knowledge-based systems.

- *Terminology* is often neglected as computationally significant. However, it is particularly important in knowledge-based systems where it is critical that the terms used in questions, recommendations, and explanations be understood by clients in the same way as they are understood by experts. Obtaining the terminology direct from experts without distortion is an important design consideration for knowledge acquisition systems.
- *Annotation* is used here as a term for noncomputational knowledge such as text, diagrams, photographs, videos, and other media, which may be used to impart knowledge to people. Some of it may be suitable for computer storage but not for inferential processing. Providing appropriate access to such knowledge is important in many knowledge-based systems, and collecting and organizing such knowledge are part of the knowledge acquisition process.
- *Classes, objects, relations, properties, and values* form the abstract infrastructure of computational knowledge representation currently. Object-oriented knowledge bases provide a formal conceptual and computational schema which subsumes and supports a wide variety of knowledge representation schema (Shriver and Wegner, 1987). These primitives also correspond to natural cognitive operations of classifying, instantiating, relating, attribut-

ing, and measuring. An important aspect of knowledge acquisition is to determine these cognitive primitives in a domain (Rappaport, 1987a).
Knowledge representation goes beyond conventional data representation largely in the variety of types of value that must be supported. In a database, a numeric field may be expected to have a clearly defined value. In a knowledge base, because knowledge is "less than truth," it may be necessary to allow for the value's being unknown, constrained by a fuzzy hedge, a range, a distribution, or some combination of these. There may also be mutual constraints between such ill-defined values.

- *Inference schemata* are systems of constraints that express the relations between values in a knowledge base. These constraints may be thought of as rules expressing the relations, and many may necessarily be expressed by explicit rules in the knowledge-base. However, the effective organization of the knowledge base in terms of generic classes and relationships should allow the majority of constraints to be expressed implicitly in a natural way. This is the *framing problem* in knowledge representation for a domain, and an important aspect of knowledge acquisition is to determine a representation schema that minimizes the number of low-level rules required.
- *Explanation schemata* are forms of argument that generate satisfactory explanations of the inferences made from a knowledge base. Early expert systems generated explanations by displaying the chain of reasoning that was used to go from premises to conclusions. This approach is useful but does not necessarily "explain" the conclusions in a way that an expert would, or in a way satisfying a client's need for knowledge. The acquisition of knowledge necessary for adequate explanation goes beyond that adequate for performance.
- *Computation, retrieval, display, communication,* and other data processing activities play major roles in most knowledge-based systems. Modern expert system shells make provision for integration with data processing, simulation, database, graphics, and communications programs. The systems analysis for these needs to be integrated with the knowledge acquisition process.
- *User interfaces* are very important in knowledge-based systems for generally they are highly interactive and involve clients with little or no computer experience. The design of the interface in consultation with experts and clients is an important part of knowledge acquisition. Since most knowledge is fairly volatile, knowledge acquisition and updating the knowledge base are important aspects of system operation. The interface to system maintainers and enhancers also needs careful attention.

The emphasis on the complexity of the social processes underlying knowledge creation, dissemination, and application in society and the need to treat knowledge-based systems as an integral part of information technology make the development of such systems appear somewhat daunting. And so it is—it is easy to underestimate the effort required to develop effective expert systems. However, knowledge acquisition for knowledge-based systems has

also tracked other trends in information technology, notably the use of the computer itself to support the system designer through computer-aided software engineering (CASE) (Gaines, 1988d). The following section describes these trends and their effects on the role of the knowledge engineer, and later sections describe and illustrate CASE tools for knowledge-based systems.

4 KNOWLEDGE ENGINEERING

Knowledge acquisition for practical system development has come to be termed *knowledge engineering,* following Feigenbaum's (1980) use of the term to describe the reduction of a large body of knowledge to a precise set of facts and rules. The term *knowledge engineer* describes the person responsible for such system development, and concise job descriptions for knowledge engineers have been given (Hayes-Roth, Waterman, and Lenat, 1983):

> 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 formalize the domain concepts.

Thus, the basic model for knowledge engineering has been that the knowledge engineer mediates between the expert and knowledge base, eliciting knowledge from the expert, encoding it for the knowledge base, and refining it in collaboration with the expert to achieve acceptable performance. Figure 3-6 shows this basic model with manual acquisition of knowledge from an expert followed by interactive application of the knowledge with multiple clients through an expert system shell:

- The knowledge engineer interviews the expert to elicit his or her knowledge.
- The knowledge engineer encodes the elicited knowledge for the knowledge base.
- The shell uses the knowledge base to make inferences about a particular case.
- Clients use the shell to obtain advice about particular cases.

Figure 3-6 may be compared with Fig. 3-4 to show the introduction of the knowledge engineer and expert system as intermediaries between the expert and client, as sociotechnical extensions to knowledge processes in society. Figure 3-1 illustrates in greater detail the many assumptions underlying the possibility of successfully introducing intermediaries in this way.

It seemed possible that knowledge engineering might develop as a profession on a par with systems analysis and programming, and that a major

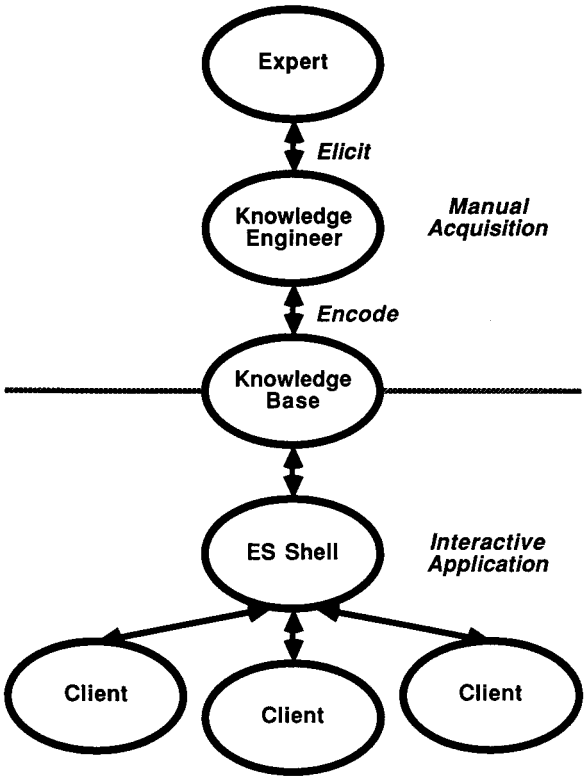

FIGURE 3-6
Basic model of knowledge acquisition by a knowledge engineer interviewing an expert.

shortage of skilled knowledge engineers would cause problems initially that would 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 major problems exist in developing applications, but doubts have been cast on the notion that human labor 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 labor in tailoring the technology for particular applications appear to be the dominating constraint and an excessive cost.
- A move toward a labor-intensive activity such as knowledge engineering is contrary to all trends in industry.

- In particular, such a move is contrary to the trend toward automatic programing 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.

These considerations aroused interest in the possibility of providing computer-based knowledge elicitation systems to automate knowledge engineering as a process of direct interaction between domain experts and the computer (Boose and Gaines, 1988). The tools and techniques used in such systems are described and illustrated in detail in later sections. Figure 3-7 shows how the role of the knowledge engineer has changed as such systems have come into use.

Interactive knowledge acquisition and encoding tools can greatly reduce the need for the knowledge engineer to act as an intermediary, but in most applications they leave a substantial role for the knowledge engineer. As shown in Fig. 3-7, knowledge engineers have responsibility for

- Advising the experts on the process of interactive knowledge elicitation
- Managing the interactive knowledge acquisition tools, setting them up appropriately
- Editing the unencoded knowledge base in collaboration with the experts
- Managing the knowledge encoding tools, setting them up appropriately
- Editing the encoded knowledge base in collaboration with the experts
- Validating the application of the knowledge base in collaboration with the experts
- Setting up the user interface in collaboration with the experts and clients
- Training the clients in the effective use of the knowledge base in collaboration with the expert by developing operational and training procedures

This use of interactive elicitation can be combined with manual elicitation and use of the interactive tools by knowledge engineers rather than, or in addition to, by experts. Knowledge engineers can directly elicit knowledge from the expert and use the interactive elicitation tools to enter knowledge into the knowledge base. Figure 3-7 shows multiple knowledge engineers since the tasks above may require the effort of more than one person, and some specialization may be appropriate. Multiple experts are also shown since it is rare for one person to have all the knowledge required, and even if this were so, comparative elicitation from multiple experts is itself a valuable knowledge elicitation technique (Boose, 1987; Shaw and Woodward, 1987; Shaw and Gaines, 1988).

Validation is discussed in greater detail in a later section. It is shown in Fig. 3-7 as a global test of the shell in operation with the knowledge base, i.e., of overall inferential performance. However, validation may also be seen as a

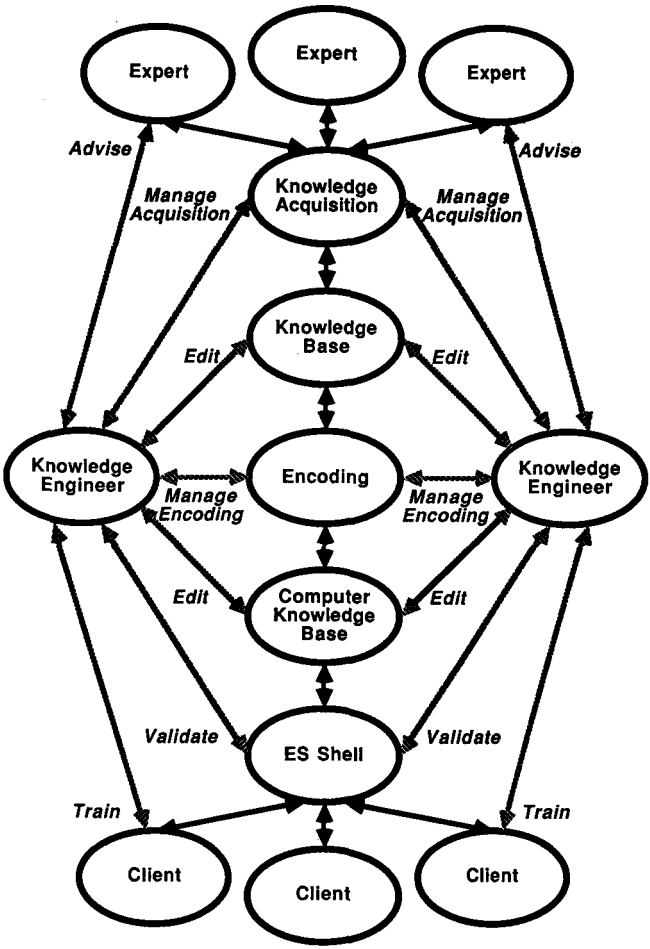

FIGURE 3-7
The knowledge engineers' roles in current system development using automated knowledge acquisition tools.

local feature of each step of the knowledge engineers' activities: the experts' proper use of the tools needs validation, the operation of the tools themselves needs validation, the resultant knowledge base needs validation, and so on. Attention to quality control through validating each step of the knowledge acquisition process is key to effective system development.

Figure 3-7 shows the complexity of the knowledge engineer's role and some of the support tools required. Figure 3-8 groups the support tools for editing, display, encoding, and validating the knowledge bases into a *knowledge-based system support environment* and lumps the various forms of knowledge base together. It shows the overall structure of a knowledge-based system as a

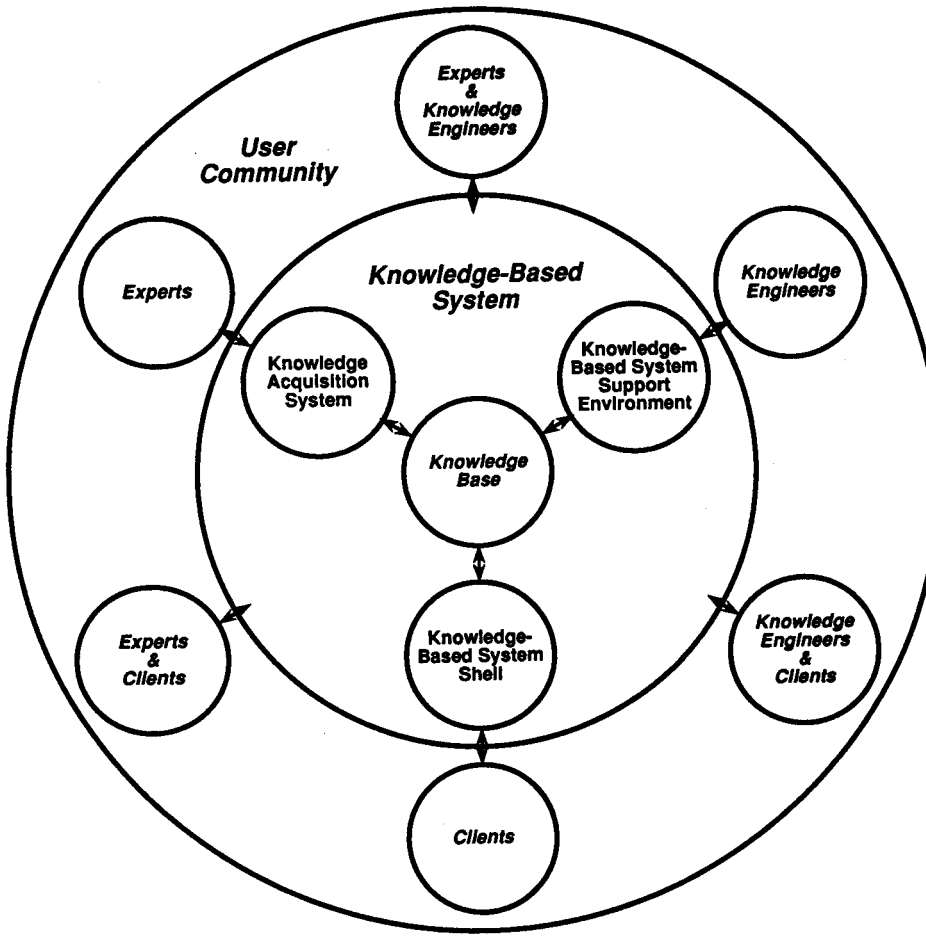

FIGURE 3-8
Major components of a knowledge-based system.

central knowledge base interacting through the knowledge acquisition tools, the expert system support environment, and the expert system shell with a user community of experts, knowledge engineers, and clients.

The notion of a *user community* is important in drawing attention to another trend in knowledge-based systems—that the distinction between experts, knowledge engineers, and clients is often one of variable roles rather than hard-and-fast job definitions. The expert may be a major client for the system or may play a major part in knowledge engineering or may do both. Some clients may customize or update the system in the role of knowledge engineers or extend its capabilities in the role of experts.

5 THE KNOWLEDGE-BASED SYSTEM SUPPORT ENVIRONMENT AND VIRTUAL MACHINE HIERARCHY

The significance of the knowledge-based system support environment shown in Fig. 3-8 in providing tools for the knowledge engineer is clear in the discussion above. The quality of this environment is very important in the development of expert systems, particularly in supporting the interaction between knowledge engineers and experts. The relation of this environment to knowledge acquisition is best seen by putting both in the context of the complete virtual machine hierarchy for knowledge-based systems. Figure 3-9 shows significant levels of this hierarchy:

1. *Machine architecture:* At the lowest level of the hierarchy is the machine on which the knowledge-based system runs. This is really irrelevant, but use of Lisp in the early days of expert systems has focused attention on Lisp machines. Now any good workstation is adequate.
2. *Operating system:* At the second level of the hierarchy is the operating system within which the implementation runs. This system needs to provide good interfaces to other programs, large databases, and communications.
3. *Implementation language:* At the third 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.
4. *Knowledge-based system shell development language:* At the fourth level of the hierarchy is the language in which the knowledge-based system shell is written, generally a special-purpose environment for coping with knowledge representation and inference.
5. *Knowledge-based system shell:* At the fifth level of the hierarchy is the expert system 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.
6. *Knowledge-based system support environment:* At the sixth level of the hierarchy is the equivalent of the application programming support environment (APSE) in conventional systems. The knowledge-based system support environment provides editing, browsing, and debugging tools for the knowledge base. This level encompasses the major differences between the high-quality knowledge-based system shells that support major system development through extensive support tools and the simpler shells that do not.
7. *Knowledge acquisition tools:* At the seventh level of the hierarchy are the tools for automating the systems analysis for expert systems, through automatic interview procedures, modeling of actual expert behavior, and analysis of knowledge in textual form. Automation of systems analysis is more significant for knowledge-based systems than for conventional data pro-

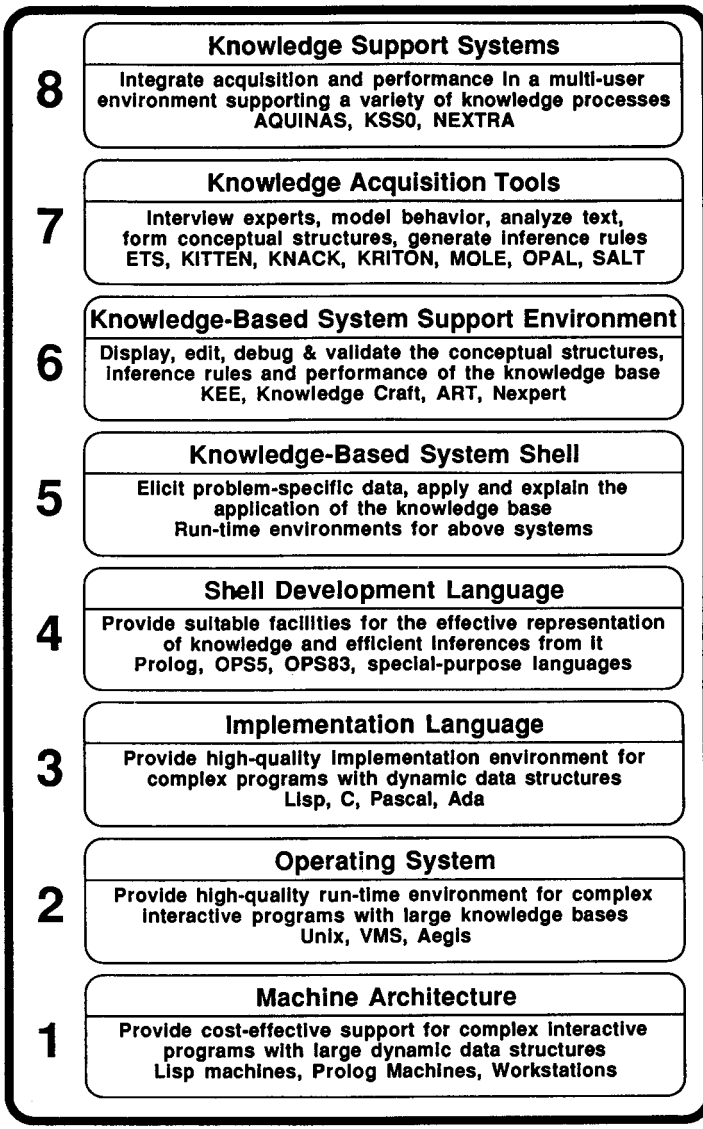

FIGURE 3-9
The virtual machine hierarchy for knowledge-based systems.

cessing because knowledge-based systems tend to be targeted on narrow areas of in-depth specialist knowledge, rather than broad general problems. In conventional terms, knowledge-based systems are satisfying the requirement for highly customized data processing based on idiosyncratic operations rather than general algorithms.

8. *Knowledge support systems:* At the top of the hierarchy are the objectives of recent research in knowledge acquisition which focuses on the integra-

tion of acquisition and performance systems to provide an environment supporting a wide variety of knowledge processes.

The hierarchy of Fig. 3-9 highlights some of the problems system managers often face in establishing development capabilities for knowledge-based systems: 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 a knowledge-based 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. But, 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.

The virtual machine hierarchy for computing is very complex and has many branches. Figure 3-9 focuses on a path through the hierarchy that links the basic hardware and software to knowledge-based systems and knowledge acquisition issues. It omits other parts of the hierarchy that illustrate the integration issues discussed previously. Figure 3-10 shows some of these issues within a hierarchical framework (Gaines, 1987a).

The split between users and technology at the top of Fig. 3-10 emphasizes the importance of effective person-computer interaction in knowledge-based systems. As with much modern information technology, usability considerations dominate effectiveness—if the interface is poor, no amount of advanced technology in the rest of the system can compensate.

The tree below this to the top left depicts the distinctions between user roles and, in particular, the tasks of the knowledge engineer already discussed. The tree to the top right first splits the technology between knowledge-based and conventional components, depicting some of the integration issues already discussed, and then further splits the knowledge-based components into support environment, shell, and acquisition.

The remainder of this chapter focuses first on the tree to the bottom left, analyzing the knowledge-based system support environment and validation, and then on the tree to the bottom right, analyzing and exemplifying the methodologies, tools, and techniques used in knowledge acquisition.

6 THE KNOWLEDGE-BASED SYSTEMS SUPPORT ENVIRONMENT AND VALIDATION

The tree on the lower left of Fig. 3-10 shows some of the major issues in the knowledge-based system support environment at level 6 in Figs. 3-8 and 3-9 in terms of the distinctions involved. The basic tools required are for editing, dis-

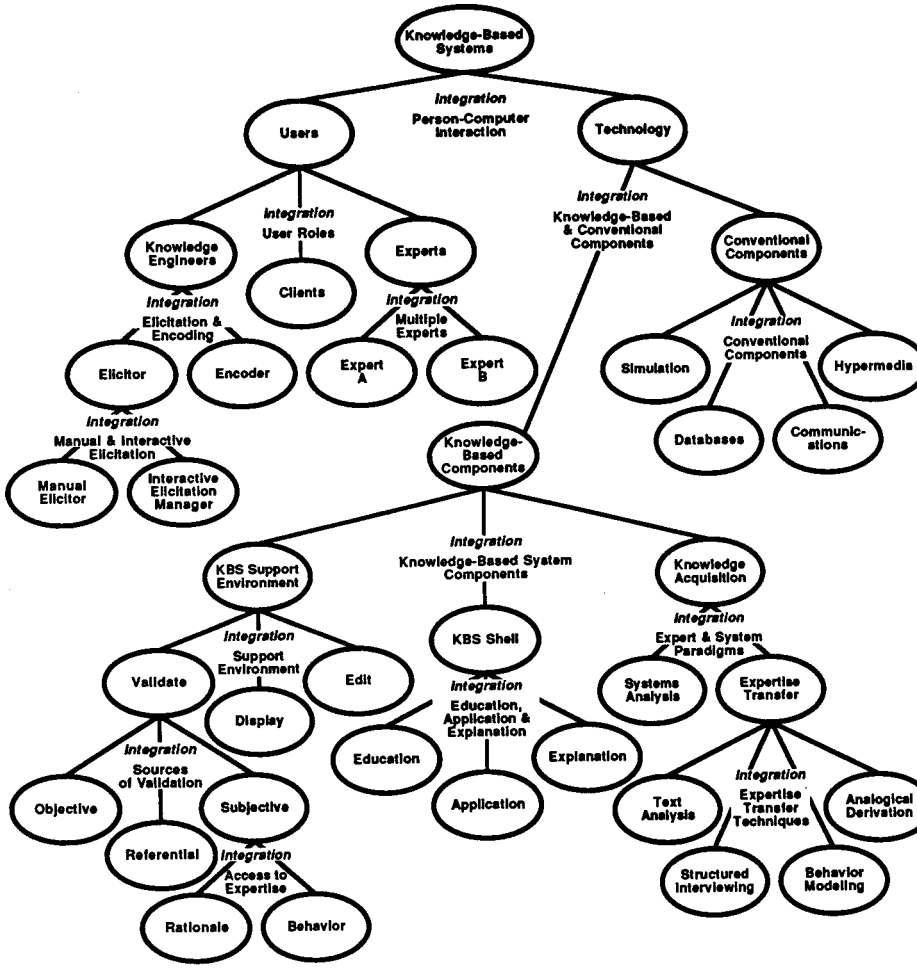

FIGURE 3-10
Integration issues in knowledge acquisition for knowledge-based systems.

playing, and validating the knowledge base. It is important that these tools present a uniform and consistent view of the knowledge base and that they aid consistency maintenance in the knowledge base during editing.

Editing is often regarded as a mundane requirement. However, the variety of related knowledge representations usually present in a knowledge-based system, particularly in the acquisition phase, makes the editing of knowledge bases very difficult to support. If representation A, e.g., text, is processed to give representation B, e.g., entities, attributes, and values, then the editing of the material in representation A should result in changes to that in representation B, and vice versa. However, the relation between representations is usually not functional but merely constrained; i.e., knowledge in representa-

tion A does not determine that in representation B, and vice versa, but only constrains it in some way, as shown in Fig. 3-11. Thus, the result of editing knowledge in one representation does not necessarily yield well-defined changes in other representations, only the detection of constraint violations. The editing system should minimally highlight the violations, but usually it can do more, e.g., it can adumbrate "likely" changes in other representations that will remove the violation.

Validation is a complex topic in its own right (Shaw and Woodward, 1987; Benbasat and Dhaliwal, 1988), and a variety of tools should be provided to support different forms of validation. The objective validation of the knowledge base in terms of its performance, its subjective validation in comparison with experts, and its referential validation in comparison with other knowledge sources should be supported. Objective validation may require significant case histories that can be collected only from an operational system in actual applications. Referential validation may require deep analysis of derivations of the knowledge base from other knowledge bases. There is a natural separation in subjective validation between experts' direct agreement that their rationale corresponds to that of the knowledge base and the observed correspondence between their behavior and that of the system based on the knowledge base. Both types of validation should be supported.

The validation of knowledge-based systems has all the dimensions of normal scientific validation together with those concerned specifically with the emulation of expertise (Gaines, 1988a). As shown in Fig. 3-12, the system can be validated through four primary pathways:

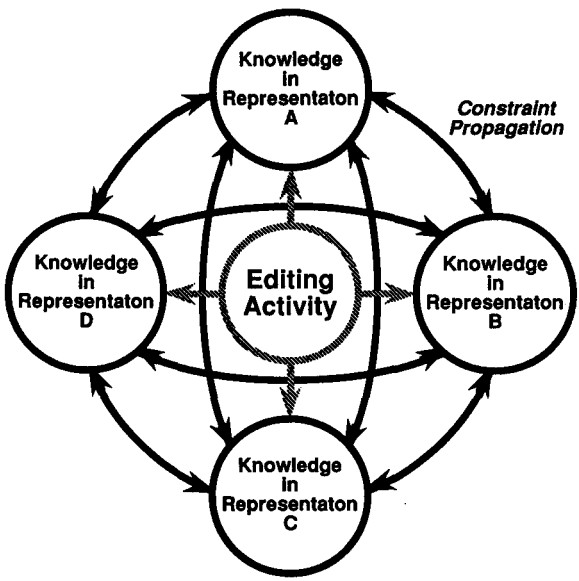

FIGURE 3-11
Constraint propagation in editing different representations of related knowledge.

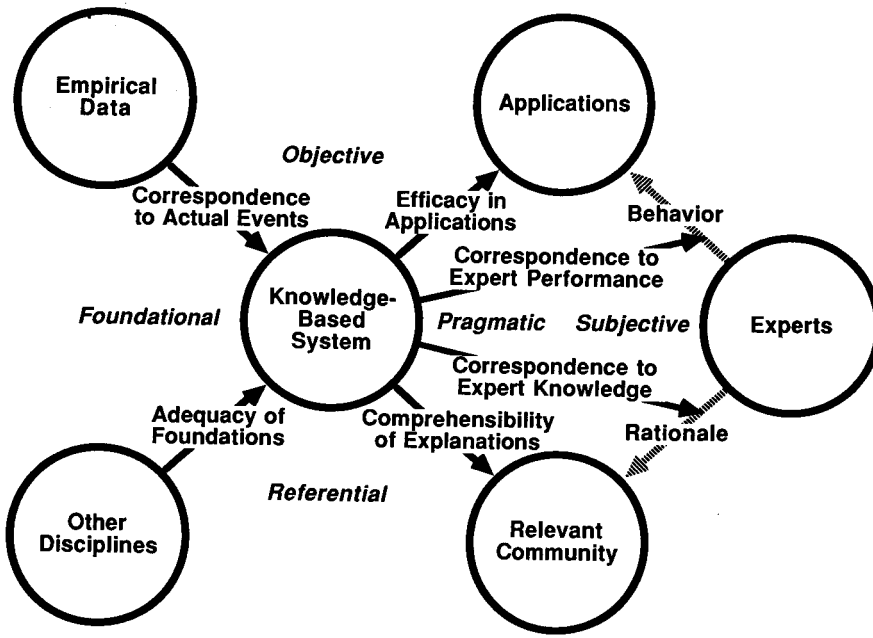

FIGURE 3-12
Dimensions of validation of knowledge-based systems.

- Through its correspondence to actual events checked against empirical data
- Through its efficacy in applications checked against actual applications
- Through the adequacy of its foundations checked against other disciplines corresponding to the derivation of presuppositions from other disciplines—this is sometimes termed *deep knowledge*
- Through the comprehensibility of explanations given checked against the relevant reference community—corresponding to explanation and communicative action

The expert systems paradigm introduces the two additional pathways shown on the right of Fig. 3-12 by taking into account the possibility that there may be "experts" who are competent in applications and explanations:

- Through its correspondence to expert performance checked against the behavior of experts in applications
- Through its correspondence to expert knowledge checked against the rationale of experts in communicating with their reference community

This diagram may be iterated a number of times to show the source of the knowledge structures of experts from empirical data and other disciplines, introducing considerations of the validity of experts and reference communities. Similarly, the diagram may be iterated to show the way in which other disciplines, in turn, are validated against their own empirical data, deeper founda-

tions, applications, and reference communities. There is no intrinsic conflict between the additional pathways of validation introduced and the positivist viewpoint of science emphasizing the primacy of empirical data. However, there is a major increase in the realism of this model in modeling the actual processes of knowledge acquisition rather than the normative criteria adopted in science as ultimate arbitrators.

7 MAJOR SOURCES FOR KNOWLEDGE ACQUISITION

The significant features of the knowledge acquisition component on the lower right of Fig. 3-10 are best analyzed by considering the natures of the classical and knowledge-based system design paradigms. Clearly knowledge-based systems use some new computational techniques, but perhaps less clear is 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 involves the instrumentation, data collection, modeling, and optimization sequence shown on the left of Fig. 3-13:

- Knowledge of past case histories is used to select a class of system models.
- The information required to discriminate within this class determines how the system should be instrumented for data acquisition.
- Data is collected from the system through the instrumentation.
- A model is identified from the model class which best fits the data.
- This model is used to design a decision or control system for optimal performance.

This approach to system design underlies the methodologies of the physical sciences and technologies based on them. 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 modeling. Its greatest successes have been where this amenability can be achieved normatively, i.e., in cases where the system to be controlled is itself a human artifact. For example, linear system theory has become a major tool in systems engineering but not because most natural systems are linear—they are not. The implication is in the opposite direction: Linear systems are mathematically tractable, and we design artificial systems to be linear so that we may model them readily.

The knowledge-based systems paradigm is particularly applicable when it is not possible to model the system but an alternative source of data is available, because human operators are able to performance the decision or control task. The right-hand column of Fig. 3-13 shows the use of knowledge sources in the expert system design paradigm:

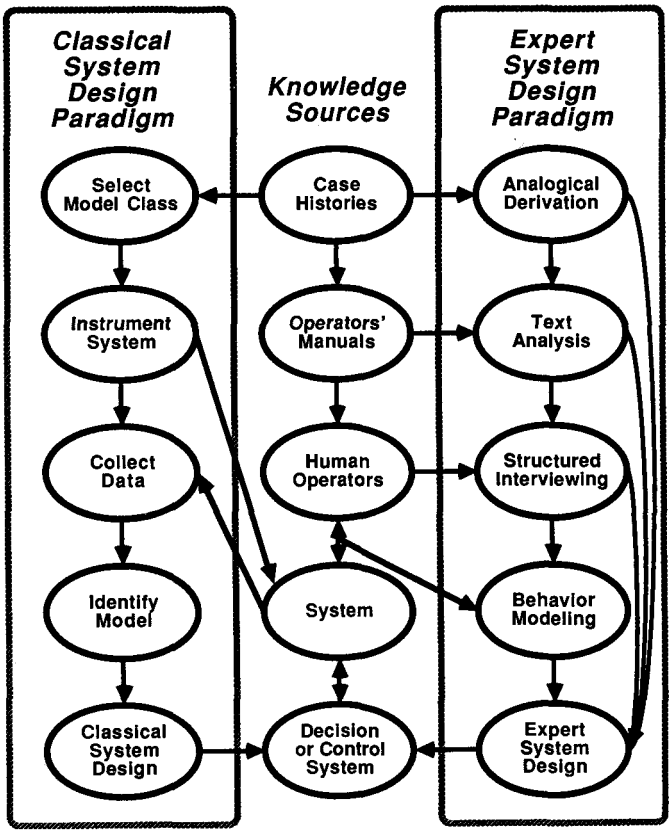

FIGURE 3-13
Classical and knowledge-based system design paradigms.

- Structured interviewing may be used to acquire knowledge directly from the operators.
- Behavior modeling may be used to identify the operators' strategies even if they are unaware of them or give incorrect ones in interviews.
- Text analysis may be used with instructional material such as the operators' manuals.
- Reasoning by analogy may be used based on the case histories without the data collection of the classical system design paradigm.

Note that the classical and knowledge-based system design paradigms, and the various acquisition techniques for knowledge-based systems, need not be regarded as competitive. They may all be used in the system design, and a collection of weak methods may be necessary because there is no overall strong method that can be applied.

The integration of the classical systems analysis with expertise transfer techniques is a major task. The primary problems currently involve the widely

different forms of knowledge representation, and these are noted in a later section in terms of the integration of conventional and knowledge-based system components. However, at a knowledge acquisition level, the problem is one of imbalance in the types of knowledge structures studied in depth to date. There has been an imbalance favoring symbolic knowledge to redress the past emphasis on quantitative methods. This bias needs redressing with the knowledge structures underlying arithmetic being integrated with those underlying nonarithmetic symbol manipulation.

8 KNOWLEDGE ACQUISITION TOOLS

A number of tools have been developed for the rapid prototyping of knowledge-based systems automating some of or all the methodologies on the right of Fig. 3-13. The objective of systems such as PLANET (Shaw and Gaines, 1983, 1986a, 1987a), ETS (Boose, 1984, 1986), MOLE (Eshelman et al., 1986), SALT (Marcus, 1987), KITTEN (Shaw and Gaines, 1987b), KNACK (Klinker et al., 1986), KRITON (Diederich, Ruhmann, and May, 1986), OPAL (Musen et al., 1986), AQUINAS (Boose and Bradshaw, 1987) and KSSO (Gaines, 1987c) is to expedite the process of knowledge acquisition and transfer to knowledge-based systems.

Boose (1988) has surveyed a very wide range of knowledge acquisition tools and classified them along a variety of dimensions, analyzing the applicability of tools to tasks. He distinguishes three classes of tasks:

1 Analysis tasks
   - **a** Classification: categorizing based on observables
   - **b** Debugging: prescribing remedies for malfunctions
   - **c** Diagnosis: inferring system malfunctions from observables
   - **d** Interpretation: inferring situation descriptions from sensor data

2 Synthesis tasks
   - **a** Configuration: configuring collections of objects under constraints in relatively small search spaces
   - **b** Design: configuring collections of objects under constraints in relatively large search spaces
   - **c** Planning: designing actions
   - **d** Scheduling: planning with strong time and/or space constraints

3 Tasks combining analysis and synthesis
   - **a** Command and control: ordering and governing overall system control
   - **b** Instruction: diagnosing, debugging, and repairing student behavior
   - **c** Monitoring: comparing observations to expected outcomes
   - **d** Prediction: inferring likely consequences of given situations
   - **e** Repair: executing plans to administer prescribed remedies

Boose notes that most knowledge acquisition tools currently are best suited to analysis tasks and are weak in acquiring expertise in synthesis.

These are the general requirements for a system to automate knowledge acquisition for knowledge-based systems:

- 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.

It is particularly important in describing tools that they be presented in the context of user interaction, since their primary function is to operate directly with experts. It is impossible to describe all the knowledge acquisition tools currently in the literature, and the references cited at the beginning of this section and specialist collections should be consulted for details and examples of use (Gaines, and Boose, 1988; Boose and Gaines, 1988; Marcus, 1988). The next section takes one particular tool that covers the range of methodologies discussed in Section 7 and uses it to illustrate the major features of a wide variety of tools.

9 EXAMPLE OF AN INTEGRATED KNOWLEDGE ACQUISITION SYSTEM

The problems of eliciting knowledge from experts as skilled performers of a task are described in Section 1. Clinical psychologists see the problem as one of cognitive defenses that impede internal communication, and they have developed techniques of verbal interaction to identify underlying cognitive processes (Freud, 1914; Kelly, 1955; Rogers, 1967). These can be used to bypass cognitive defenses, including those resulting from automatization of skilled behavior, and Welbank (1983) has surveyed psychological techniques for doing this. In an industrial context, the application of techniques derived from clinical psychology to problems of hard system 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 discussed above. However, for the reasons given above, this is not a realistic approach, and the focus shifted to computer-based tools, many of which have been derived from techniques of clinical and social psychology.

One major class of tools for knowledge acquisition has been developed around the "personal construct psychology" of George Kelly (1955) and, in particular, his repertory grid technique for eliciting conceptual systems as individual models of the world. In computational terms, the grid may be seen as small entity-attribute database in which a domain is characterized by an expert in terms of critical objects within it and the properties of these objects relevant to problem-solving within the domain. The elicitation of an expert's conceptual system necessarily involves approximation since a complete system may involve indefinitely complex relations. However, as a start, one wishes to elicit the major distinctions that an individual uses in a domain, the terminology used for them, and the relation of such distinctions and terminology to those of others. Entity-attribute grid elicitation is an effective method for eliciting major distinctions and terminology in a domain (Shaw, 1981).

The grid technique is an extensional approach in which individuals are asked to specify a set of entities in a domain and then make distinctions among them, naming the distinctions and classifying all the specified entities in terms of them. The extension of a distinction determined in this way is only an approximation to the underlying concept since critical entities may be missing in the classification. However, both manual and computer-based elicitation techniques attempt to prompt the individual for additional entities to discriminate between extensionally related distinctions (i.e., making the same, or similar, classifications). Knowledge acquisition is essentially a negotiation process leading to approximations to conceptual structures that are adequate for some practical purposes.

The application of interactive repertory grid elicitation to expert system development was first proposed by Gaines and Shaw (1980), taken up and demonstrated effective in industrial applications by Boose (1984), and is now in widespread use through a number of derivative system developments (Gammack and Young, 1985; Diederich, Ruhmann, and May, 1986; Wahl, 1986; Garg-Janardan and Salvendy, 1987). Figure 3-14 gives a family tree for some of the major developments in repertory grid knowledge acquisition tools and the extensions of Kelly's original concepts.

KSS0: Knowledge Support System Zero

Figure 3-15 shows the architecture of one system that integrates a wide range of knowledge acquisition methodologies. KSS0 is written in Pascal and runs on the Apple Macintosh family of computers to provide a highly interactive and graphical knowledge acquisition environment. At the heart of KSS0 is an object-oriented knowledge base in which knowledge is formally represented as a multiple-inheritance structure of classes, objects, properties, values, and re-

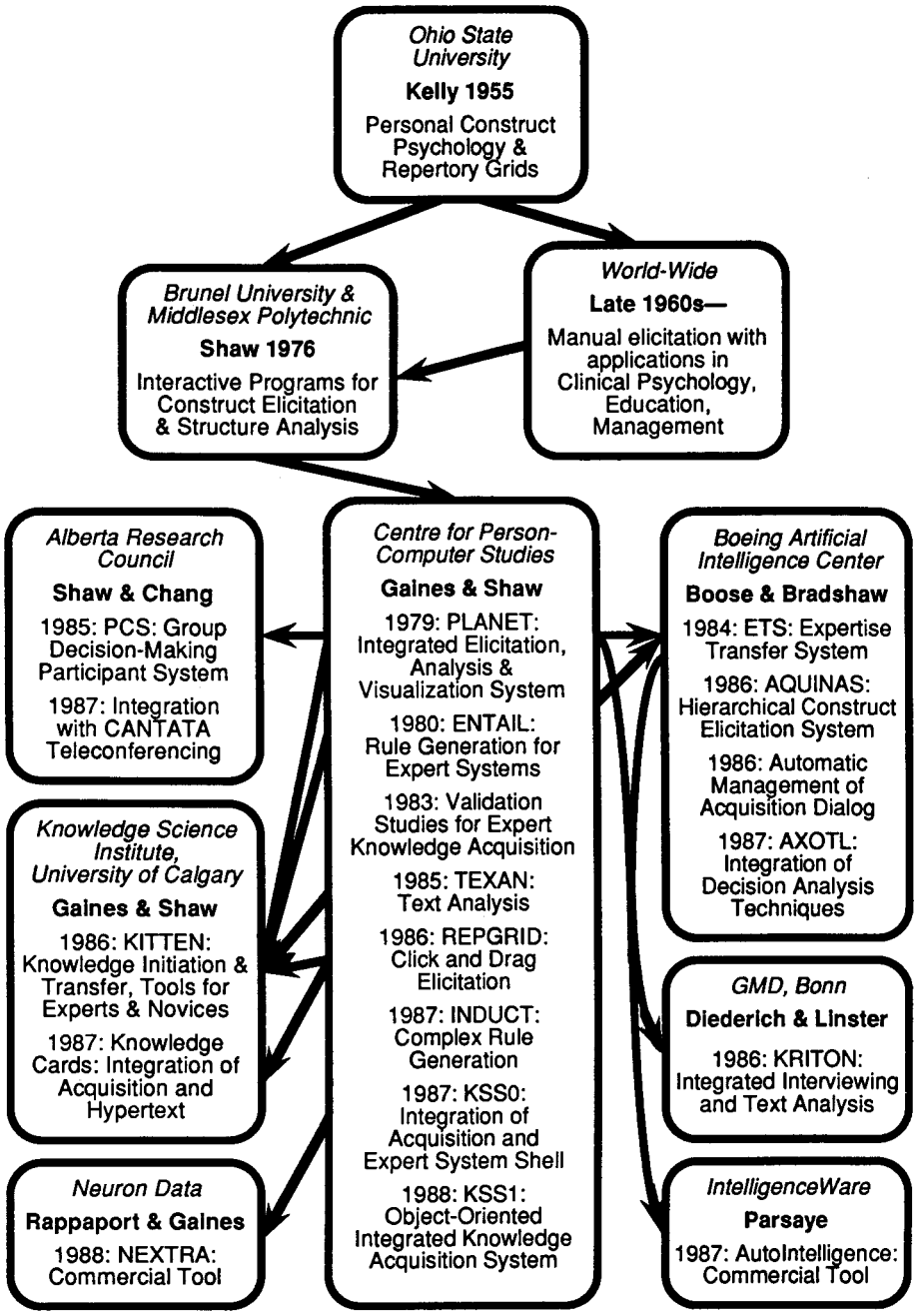

FIGURE 3-14
Family tree of applications and developments of repertory grids in knowledge acquisition.

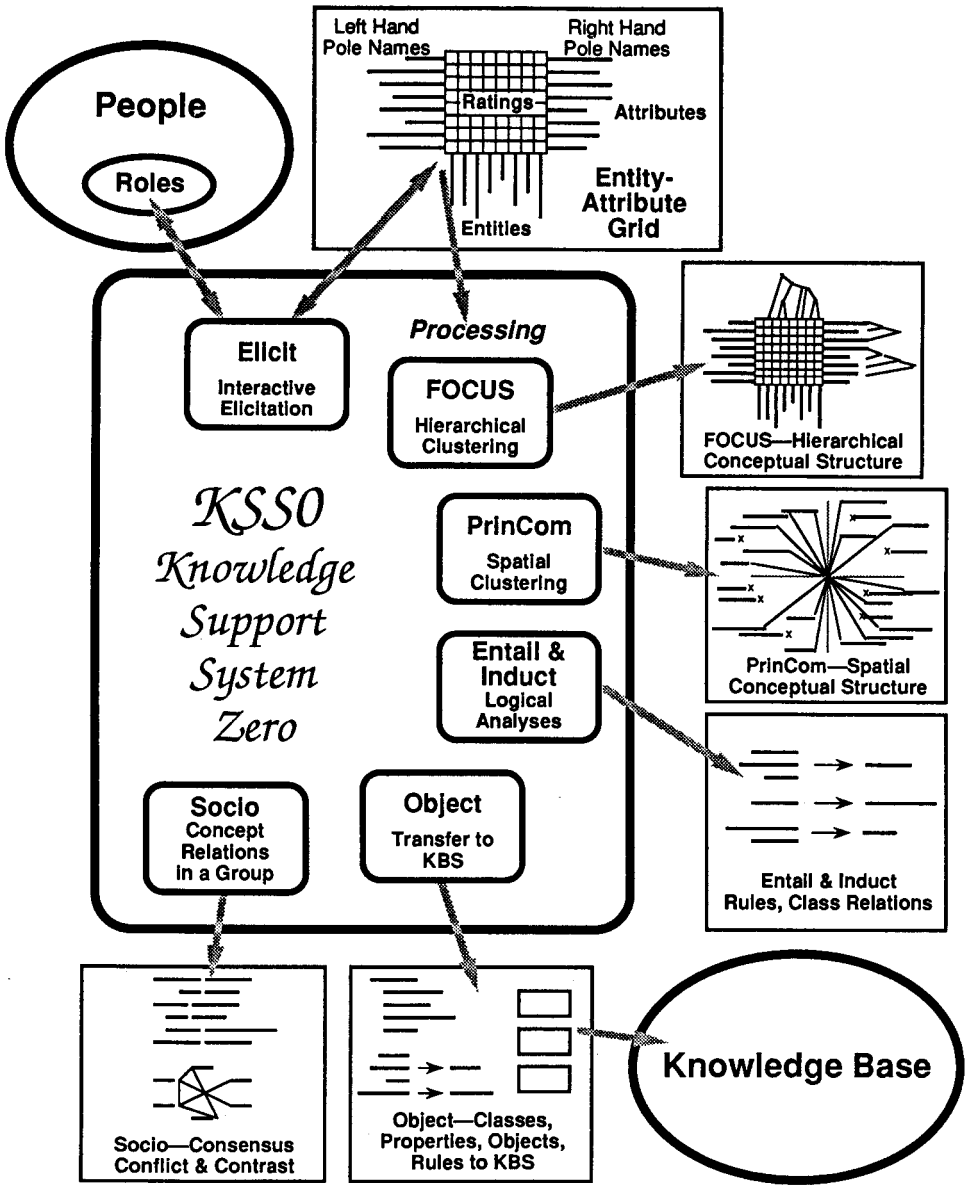

FIGURE 3-15
KSS0, an integrated knowledge acquisition and support system.

lations. Such a structure generalizes the entity-attribute grids used in several early knowledge acquisition systems and has proved both general and powerful in a variety of applications.

The elicitation tools in KSS0 are based on Shaw's (1980) computer-based interviewing techniques extended through the use of graphical rather than nu-

merical data entry. The visualization and the direct manipulation of knowledge structures through the graphical and click-and-drag facilities of modern workstations are an important development at the user interface giving experts, knowledge engineers, and clients improved access to the knowledge structures.

- *Interview* accepts specifications of entities within a subdomain and provides an interactive graphical elicitation environment within which the experts can distinguish entities to derive their attributes. The resultant class is continuously analyzed to provide feedback, prompting the expert to enter further entities and attributes.

The visual analysis tools consist of an interactive interface, to represent the abstractions derived from those entities in terms of hierarchical clusters using Shaw's (1980) FOCUS algorithm, and relational diagrams such as nonhierarchical conceptual maps derived through principal components analysis (Slater, 1976). The objectives are to validate the raw domain knowledge and to suggest further structure at a higher level through interactive topological induction (Rappaport and Gaines, 1988):

- *FOCUS* hierarchically clusters entities and attributes within a subdomain, prompting the experts to add higher-level entities structuring the domain.
- *PrinCom* spatially clusters entities and attributes within a subdomain, prompting the experts to add higher-level entities structuring the domain.

The group comparison tools consist of an interactive interface to represent the relations between the terminologies and conceptual systems of different experts, or experts and clients. The objective is to determine the consensus, conflict, correspondence, and contrast between different conceptual systems (Shaw and Gaines, 1988):

- *Socio* compares the structures for the same subdomain generated by different experts or by the same expert at different times or from varying perspectives.

The inductive part consists in the derivation of constraints within the conceptual structures through logical entailment analyses (Gaines and Shaw, 1985; Quinlan, 1987; Cendrowska, 1987). The objective is to suggest further structure at a higher level that translates to class inclusions or rules in NEXPERT:

- *Entail* and *Induct* induce logical entailments enabling the attributes of an entity, or the evaluations of a decision-making situation in a domain, to be derived from other attributes.

The generative part consists in the transformation of the knowledge analysis made by the previous tools into a formalism understandable by the NEXPERT inference mechanisms (Rappaport, 1987b):

- *Object* formats the specifications of subdomains as classes, of entities as objects, of attributes as properties, and of entailments as methods, and it transfers them to the performance tool.

Knowledge Acquisition in Action: Defining the Domain

Figure 3-16 is the start-up screen for KSS0. The system is designed to be very clean and simple to use with a substantial entertainment value. The objective is to make interaction with system attractive and fun while retaining a professional working environment. Help is available at all times through context-sensitive, interactive "advice buttons." The expert should never be at a loss, need help that she or 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 through biased 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 3-17 is an entity screen from KSS0 in a knowledge elicitation sequence concerned with factory layout planning (Adler and 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—

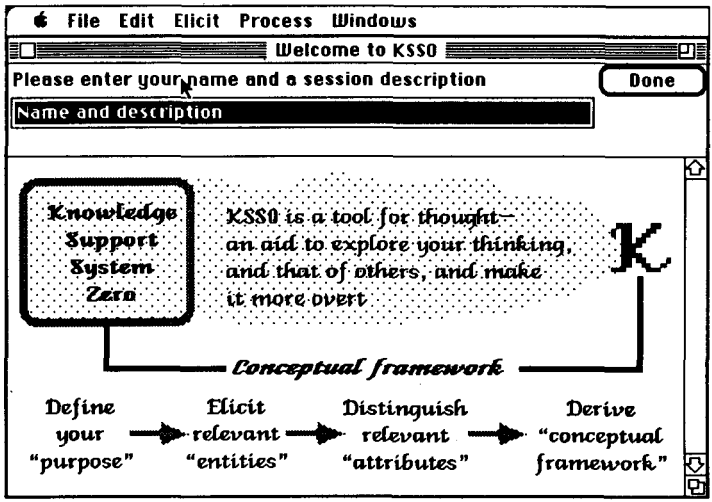

FIGURE 3-16
KSS0 start-up screen.

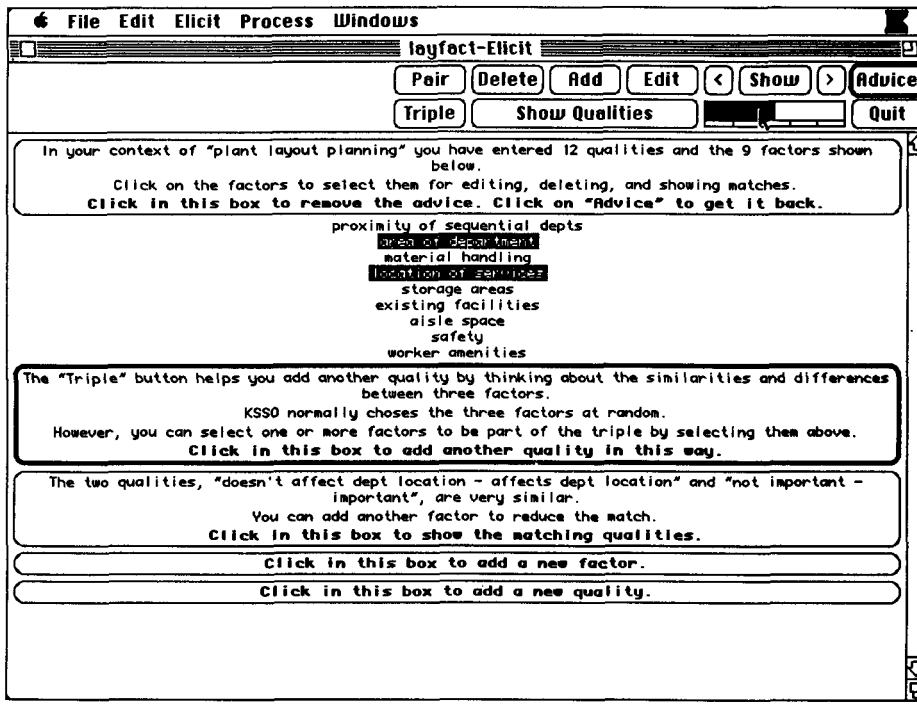

FIGURE 3-17
Entity screen showing matches.

these are the entities—and we will see the system help the expert to elicit relevant "qualities"—these are the attributes.

The interactive advice buttons are also apparent in this figure. They are generated dynamically from the current context and may be used to select actions without use of the control panel above, or the pull-down menus. "Power users" may turn them off completely to allocate more screen space to the data structures—this works well because by the time the user has entered a substantial number of entities and attributes, she or he has become familiar with the system and no longer needs the tutorial advice.

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 upper right of Fig. 3-17 shows the value of the highest match between entities. The expert has clicked on it to highlight the matching entities and can use the " < " and " > " buttons to step backward and forward through the matches.

The expert next clicks on the "Show" button to see whether he or she can input an attribute that will reduce the match, i.e., a counterexample to the inferred structure. KSS0 goes to the screen shown in Fig. 3-18 which makes explicit the question that needs to be answered and shows 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) shown by the upward markers and those of the lower entity (area of department) 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 the user 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 the action by answering yes to the question asked. KSS0 then goes to the screen shown in Fig. 3-19 where the expert 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 3-20 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. Then the expert clicks on the "Show" button again and moves to the screen shown in Fig. 3-21, 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. The expert can again edit the ratings by clicking

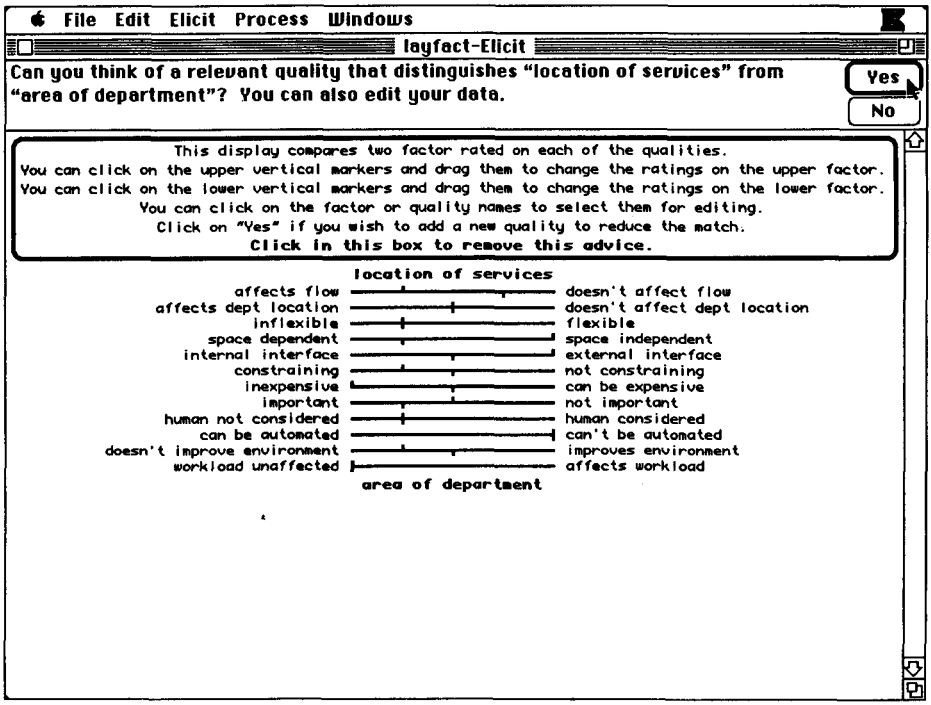

FIGURE 3-18
Entity match screen.

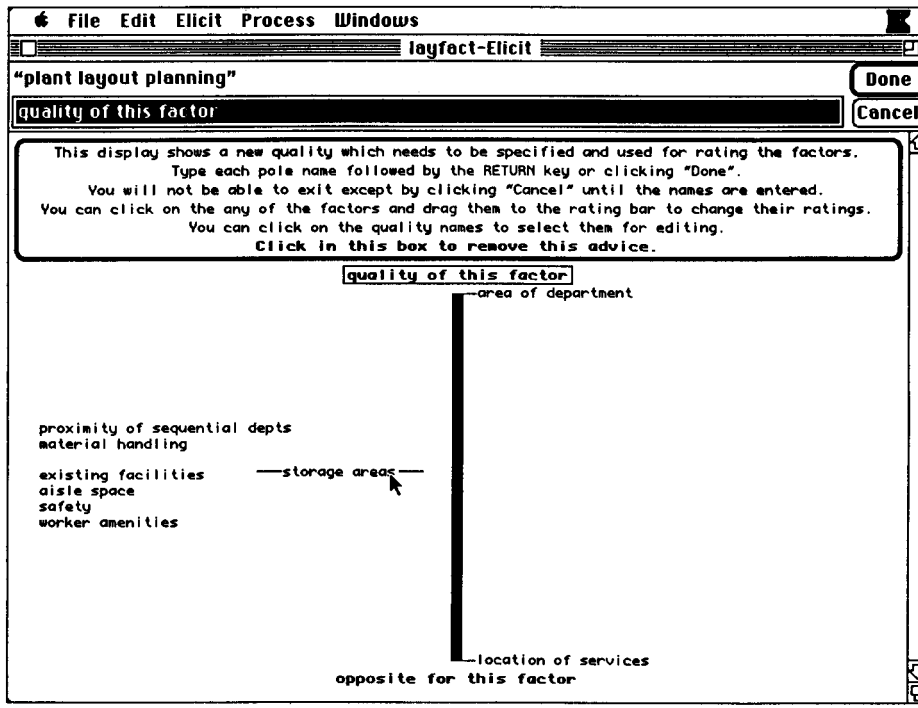

FIGURE 3-19
Attribute rating screen.

and dragging the entities if he or she wishes. Answering yes to this question takes her or him to the screen in Fig. 3-22 where he or she can enter the name of the new entity and then rate it on each of the existing attributes.

Figure 3-23 shows a different form of attribute elicitation initiated when the expert clicks on the "Triple" button in Fig. 3-17. KSS0 generates a triple of three entities and asks in what way two are alike and differ from the third. The expert clicks on "storage areas," and KSS0 goes to the rating screen shown in Fig. 3-24 with the triple prerated, so that the expert may enter the attribute and the ratings of the other entities. The triples are generated on a random basis biased to promote variety, and this mode of elicitation tends to trigger recollections of significant attributes otherwise forgotten.

Knowledge Acquisition in Action: Visualization

The elicitation screens shown above all use the graphical capabilities of the workstation to encourage and support the experts' visualization of the data being entered. The rating of an entity on an attribute is seen in relation to the ratings of other entities to enable immediate consistency checks to be made. The analysis tools continue this emphasis on visualization and validation by

presenting the conceptual structures underlying the data entered in graphical form.

At any time during the elicitation, experts can ask for the structure of their entity-attribute conceptual system to be analyzed and displayed. Figure 3-25 shows the basic data structure of the grid that has been entered with entity names, attribute names, rating values, and shading to indicate low, medium, and high values. Figure 3-26 shows a hierarchical clustering of the entities and attributes in Fig. 3-25 based on the matches between them by using Shaw's (1980) FOCUS algorithm. Note how this has tended to pull similarly shaded areas together. The clusters can be used to suggest the introduction of hierarchical relations in the conceptual framework, e.g., that location of services, area of department, and aisle space are all spatial constraints.

Figure 3-27 shows an alternative mapping into two dimensions based on a principal components analysis (Slater, 1976). Again this can be used to identify groupings leading to more abstract attributes. Both forms of clustering are also important in allowing the expert to examine the semantic consistency of the data that has been entered—if the way that entities and attributes cluster does not make sense, then it is a clear indication that something is wrong in the data entered or the expert's conceptualization of the domain.

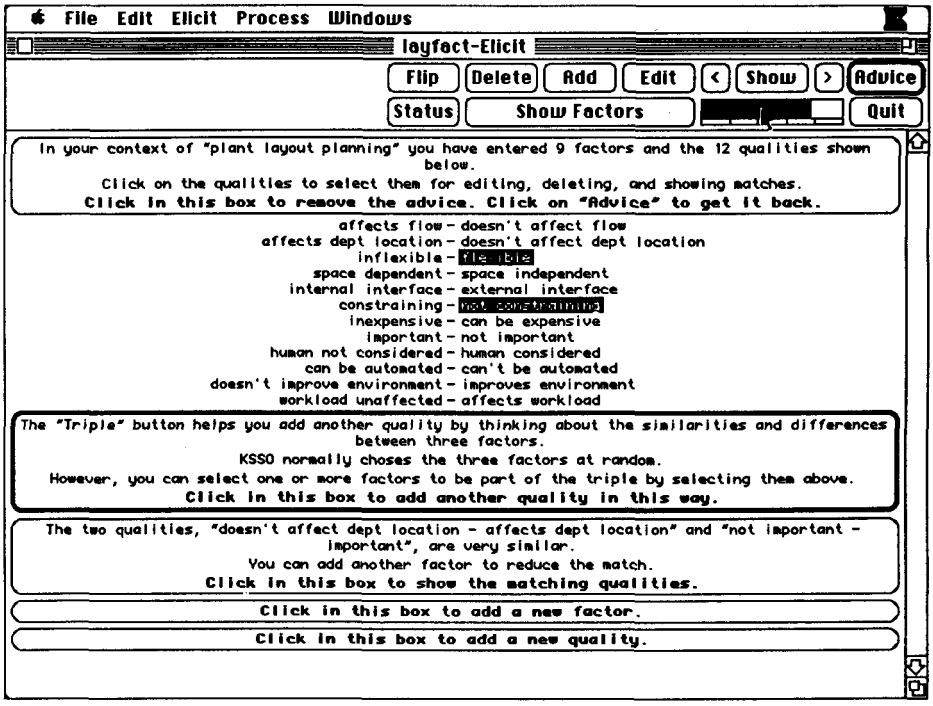

FIGURE 3-20
Attribute screen showing matches.

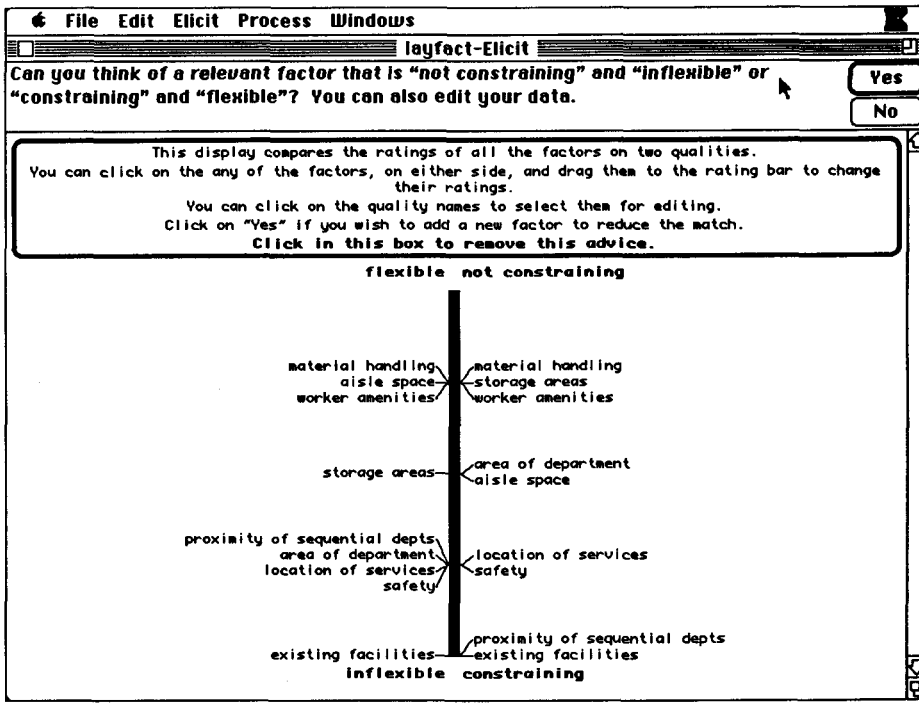

FIGURE 3-21
Attribute match screen.

Knowledge Acquisition in Action: Transfer to the Shell

Rules relating attributes are generated from the rule induction algorithms Entail and Induct applied to the grid data. The dataset elicited is small so that the statistical significance of the induced rules cannot be expected to be high. However, sorting by statistical significance is still effective in preventing the generation of spurious rules with inadequate support. The small dataset is usually adequate because all the cases have been selected by the expert as being significant and correct—it is not a random collection of noisy data.

The Object tool in KSS0 converts attribute sets to class definitions, entities to objects, and entailments to rules and then formats them as a NEXPERT knowledge base. Figure 3-28 shows the class, objects, and properties within the NEXPERT graphical editing environment, and Fig. 3-29 shows some of the rules. If the task might be solved by a decision tree, then the result is a complete, operational expert system. A more complex expert system development will involve a number of elicitations with KSS0 to develop the conceptual structures for the subdomains followed by the use of NEXPERT editing tools to link the subdomains through appropriate control structures.

The technical details of the processes going on in Figs. 3-16 through 3-28 are less important than their 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 users into an introspective frame of mind conducive to knowledge elicitation. Most of the dialogue that users 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 to the structures generated for use in the expert system shell—the structures 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 knowledge structures, such as those shown in Figs. 3-27 and 3-28, are comprehensible to the expert in their derivation and terminology. Also by this stage the expert has become accustomed to the use of the computer—windows, menus, and mice are familiar tools—and will often move over to editing the rules in the shell without really being aware 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 islands tend to form a hierarchy; e.g., "safety"

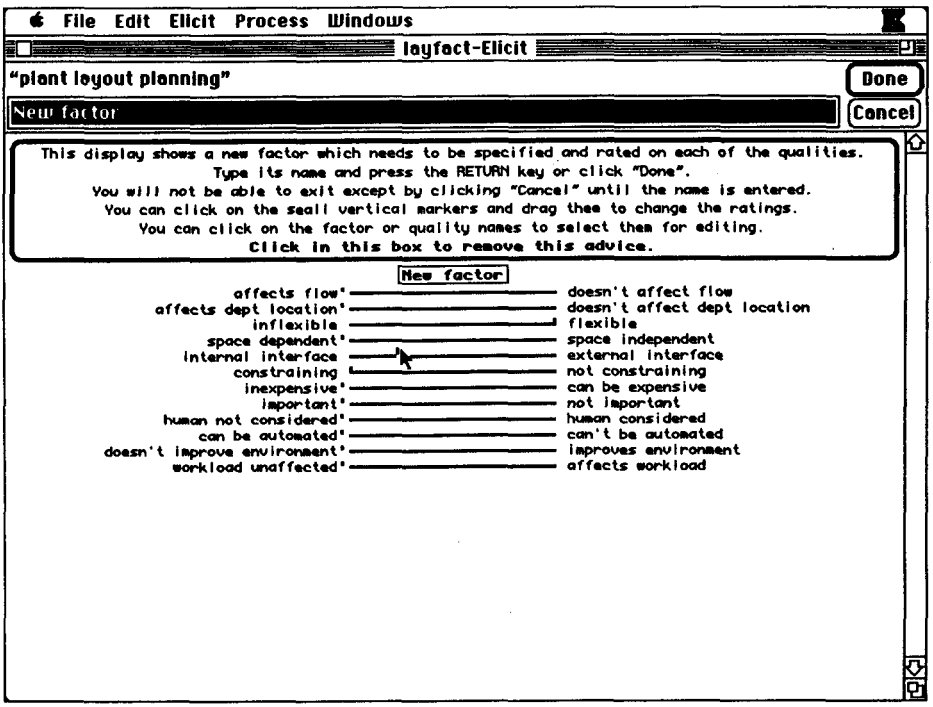

FIGURE 3-22
Entity rating screen.

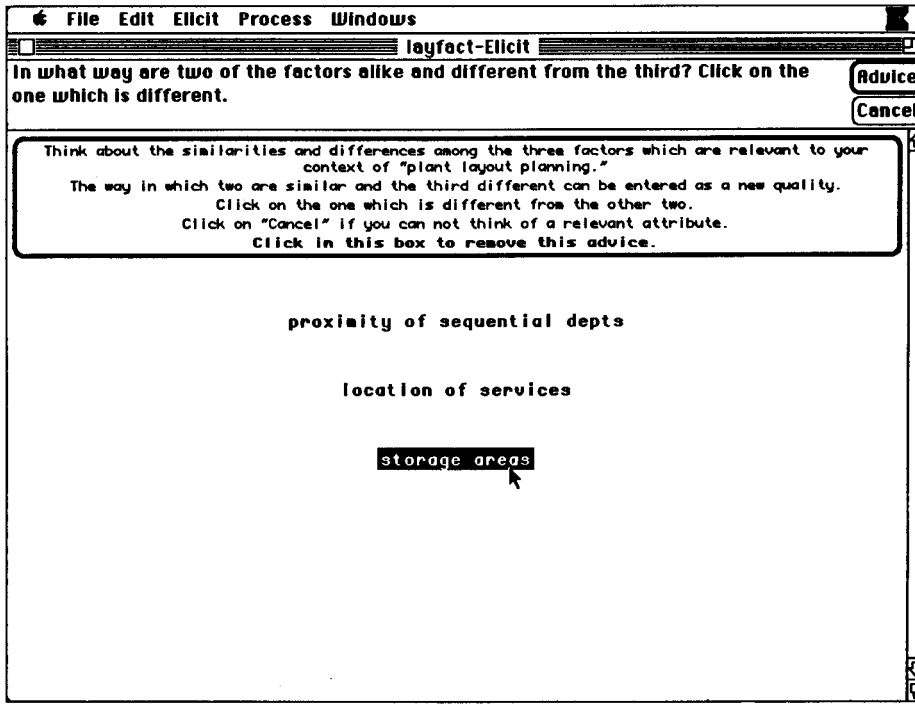

FIGURE 3-23
Triple elicitation screen.

is a general factor in the elicitation of Fig. 3-17 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 and 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 Socio tool in KSS0 allows grids elicited in the same domain from multiple experts to be compared and consensual, conflicting, corresponding, and contrasting relations between attributes to be displayed in graphical form (Shaw and Gaines, 1988).

10 CONCLUSIONS: IMPEDIMENTS TO KNOWLEDGE-BASED SYSTEM DEVELOPMENT

Expert systems technology, like many other computing developments, has been "oversold," and the expectations of what may be achieved through the

use of knowledge-based system technology have been raised too high. The bottleneck of knowledge engineering discussed in Section 4 still remains. The quotation at the head of that section dates from 1983; meanwhile major effort has swung into research on knowledge acquisition techniques, and there has been a substantial increase in the experience of developing expert systems. Notwithstanding this research and experience, there are still major problems in the development of knowledge-based systems, and it is appropriate to conclude this chapter by summarizing these problems. Figure 3-30 gives a synopsis of the major impediments to knowledge-based system development. The top box in Fig. 3-30 summarizes what may be termed the problem of *professionalism* in developing knowledge-based systems:

- *Theory:* The lack of scientific and engineering foundations for knowledge-based systems is a major impediment to the development of professional standards for expert system development. We understand remarkably little about the nature of expertise even though it is 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 human thought processes, used in human problem solving, and transmitted between people.

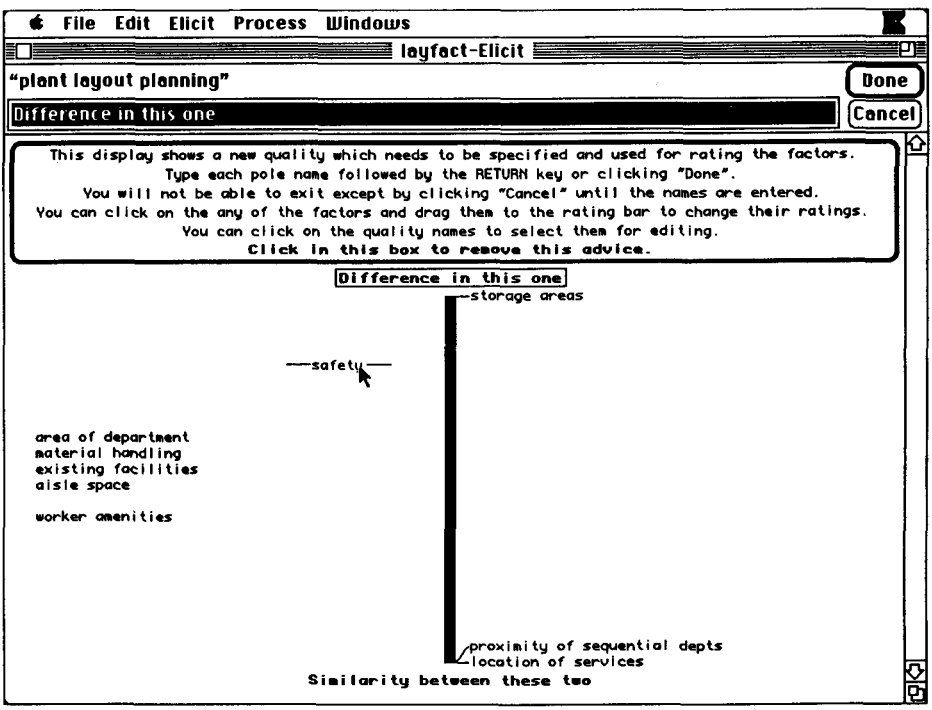

FIGURE 3-24
Attribute entry and rating after a triple.

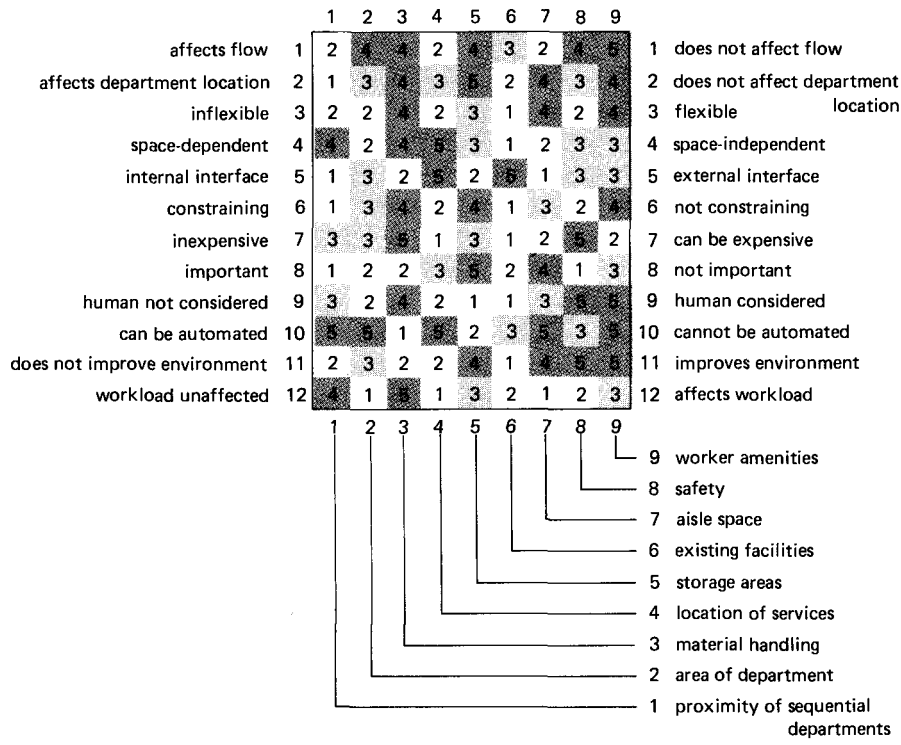

FIGURE 3-25
An entity-attribute grid.

- *Techniques:* The lack of well-established techniques for knowledge engineering is a major problem for new development teams. There is some folklore, generally mutually contradictory, and 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 major problem. There are now many research reports of such tools, but few of those developed are available as 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 are being offered by commercial and university

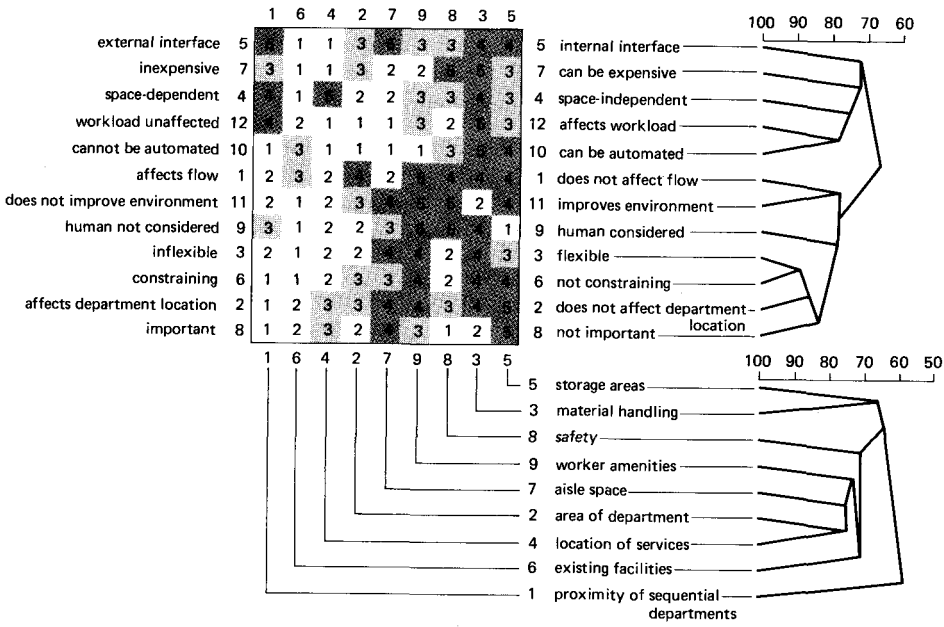

FIGURE 3-26
Hierarchical cluster analysis.

organizations. It is still a problem to get good-quality courses targeted to specific requirements and on the schedules needed.

The middle box in Fig. 3-30 summarizes problems of *knowledge acquisition and transfer* in developing knowledge-based systems. Knowledge acquisition and transfer processes in human society are complex and poorly understood. Neurology, psychology, linguistics, education, sociology, anthropology, philosophy, and systems theory all make significant contributions. However, integrating them, clarifying jargon, and combining different objectives and perspectives are major problems. A variety of short-term problems arise from these weaknesses in foundations:

- *System specification:* The formal specification of software that has made software engineering a professional discipline which can ensure quality within well-defined budgets and schedules is notably lacking for expert system developments. We seem to be at the stage of conventional software development of twenty years ago. This should change as expert system developments move out of the laboratories into the hands of experienced 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.

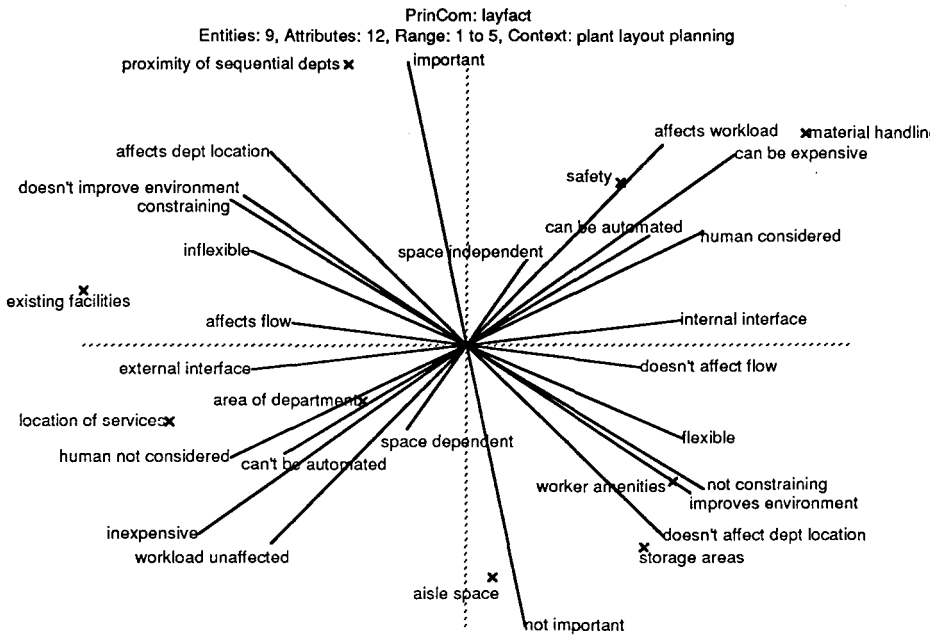

FIGURE 3-27
Spatial cluster analysis.

- *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.
- *Acquisition of skills:* Communicating the basis of skilled performance between people, and between people and computers, is very difficult.
- *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.

The bottom box in Fig. 3-30 summarizes problems of *system performance* in developing knowledge-based systems. 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. There are 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? Very few validation studies exist 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 usability, 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 misconception that access to encoded expertise can make experts of us all. 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

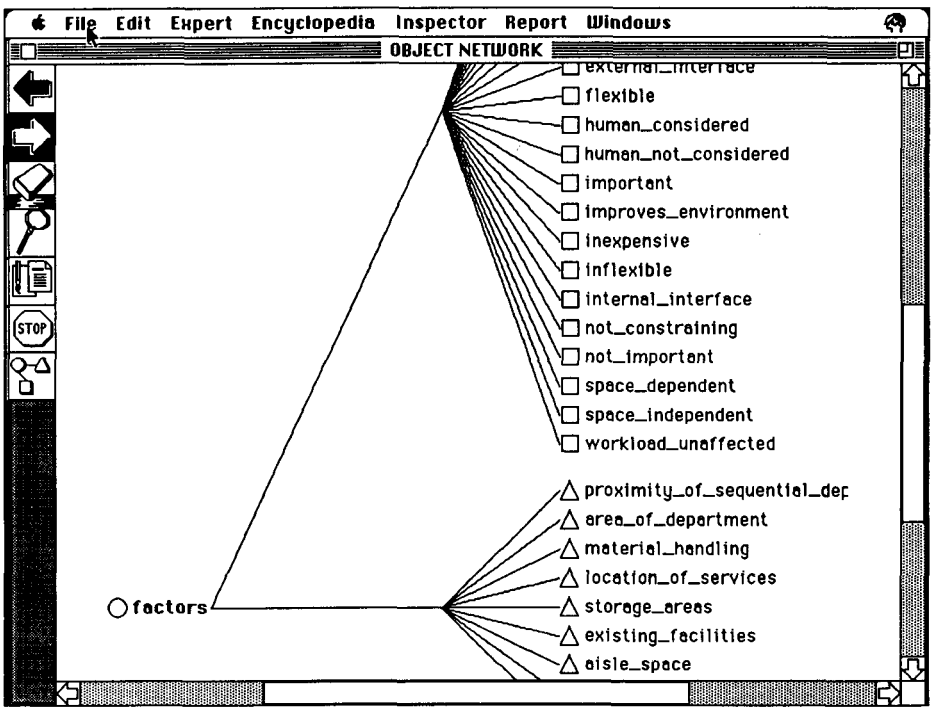

FIGURE 3-28
Class, objects, and properties exported to NEXPERT.

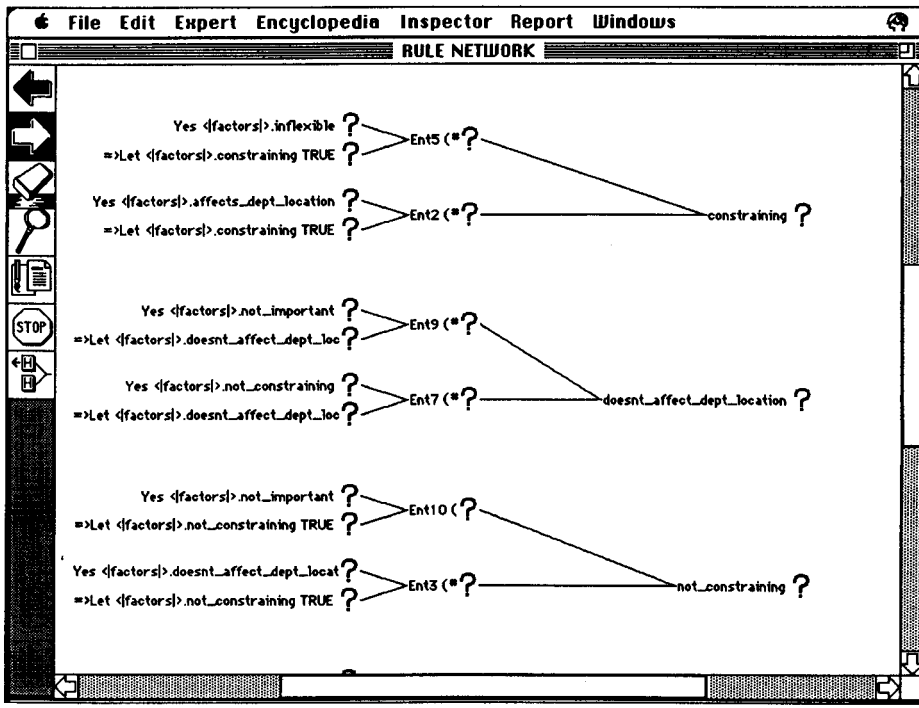

FIGURE 3-29
Rules exported to NEXPERT.

processing. The usual problems of extensive maintenance 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.

This emphasis on problems impeding knowledge-based system development is not intended to be negative. In any major technological advance there are significant problems, and it is the recognition and classification of these that indicates the growing maturity of the field. There is intensive, worldwide research in all the problem areas noted above and growing experience in how to overcome them. By recognizing the problems, project managers may be able to target their objectives to avoid some and to allocate their resources to overcome others.

ACKNOWLEDGMENTS

I am grateful to many colleagues for discussions over the years that have influenced this chapter. In particular, I would like to thank John Boose, Jeff

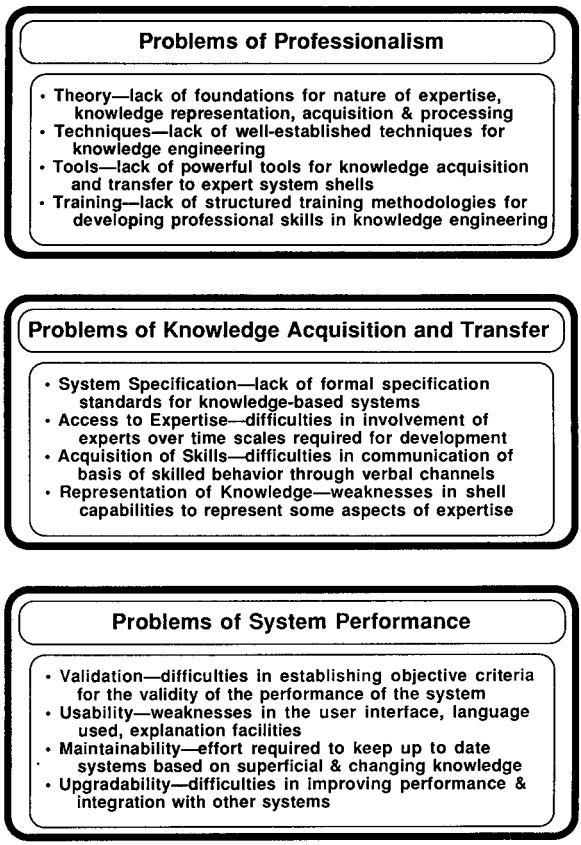

FIGURE 3-30
Impediments to knowledge-based system development.

Bradshaw, Alain Rappaport, and Mildred Shaw for access to their own research and many stimulating discussions. Much of the research reported here is part of a long-term collaborative study with Mildred Shaw of the cognitive foundations of human knowledge processes.

REFERENCES

Adler, J., and J. Matheson (1986). Expert knowledge elicitation using personal construct theory: Obtaining plant layout expertise with "PLANET." B.A. Sc. thesis, Department of Industrial Engineering, University of Toronto./ 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.

Benbasat, I., and J. S. Dhaliwal (1988). A framework for the validation of knowledge acquisition. *Proceedings of the Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop*, pp. 1-1–1-18. Banff (November).

Blume, S. S. (1974). *Towards a Political Sociology of Science*. New York: Free Press.

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

Boose, J. H. (1986). *Expertise Transfer for Expert System Design.* New York: Elsevier.

Boose, J. H. (1987). Rapid acquisition and combination of knowledge from multiple experts in the same domain. *Future Computing Systems* 1(2):191–216.

Boose, J. H. (1988). A survey of knowledge acquisition techniques and tools. In: J. H. Boose and B. R. Gaines (Eds.), *Proceedings of the Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 3-1–3-23. Banff (November).

Boose, J. H., and J. M. Bradshaw. Expertise transfer and complex problems: Using AQUINAS as a knowledge acquisition workbench for knowledge-based systems. *International Journal of Man-Machine Studies* 26:3–28.

Boose, J. H., and B. R. Gaines (Eds.) (1988). *Knowledge Acquisition Tools for Expert Systems.* London: Academic Press.

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

Cendrowska, J. (1987). An algorithm for inducing modular rules. *International Journal of Man-Machine Studies* 27(4):349–370 (October).

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

Diederich, J., I. Ruhmann, and M. May (1986). KRITON: A knowledge acquisition tool for expert systems. In: J. H. Boose, and B. R. Gaines (Eds.), *Proceedings of the First AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 12-0–12-12. Banff (November).

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

Eshelman, L., D. Ehret, J. McDermott, and M. Tan (1986). MOLE: A tenacious knowledge acquisition tool. In: J. H. Boose and B. R. Gaines (Eds.), *Proceedings of the First AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 13-0–13-12. Banff (November).

Feigenbaum, E. A. (1980). Knowledge engineering: The applied side of artificial intelligence. Report STAN-CS-80-812, Department of Computer Science, Stanford University.

Feyerabend, P. (1975). *Against Method.* London: NLB.

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

Gaines, B. R. (1987a). Advanced expert system support environments. In: J. H. Boose and B. R. Gaines (Eds.), *Proceedings of the Second AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 8-0–8-14. Banff (October).

Gaines, B. R. (1987b). An overview of knowledge acquisition and transfer. *International Journal of Man-Machine Studies* 26(4):453–472 (April).

Gaines, B. R. (1987c). Rapid prototyping for expert systems. In: M. D. Oliff (Ed.), *Intelligent Manufacturing: Proceedings from First International Conference on Expert Systems and the Leading Edge in Production Planning and Control,* pp. 45–73. Menlo Park, Calif.: Benjamin Cummins.

Gaines, B. R. (1988a). Knowledge acquisition and technology. In: J. H. Boose and B. R. Gaines (Eds.), *Proceedings of the Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 8-1–8-21. Banff (November).

Gaines, B. R. (1988b). Positive feedback processes underlying the formation of expertise. *IEEE Transactions on Systems, Man and Cybernetics,* to appear.

Gaines, B. R. (1988c). Second generation knowledge-acquisition systems. *Proceedings of the Second European Workshop on Knowledge Acquisition for Knowledge-Based Systems (EKAW 88).* GMD-Studien Nr. 143. pp. 17-1–17-14. Bonn, Germany: Gellsellschaft für Mathematik und Datenverarbeitung (June).

Gaines, B. R. (1988d). Software engineering for knowledge-based system. *Proceedings of CASE 88, Second International Workshop on Computer-Aided Software Engineering,* pp. 14-1–14-7.

Gaines, B. R., and J. H. Boose (Eds.) (1988). *Knowledge Acquisition for Knowledge-Based Systems.* London: Academic Press.

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

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

Gaines, B. R., and J. N. Vickers (1988). Design considerations for hypermedia systems. *Microcomputers for Information Management* 5(1):1–27 (March).

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

Garg-Janardan, C., and G. Salvendy (1987). A conceptual framework for knowledge elicitation. *International Journal of Man-Machine Studies* 26(4):521–531 (April).

Hagstrom, W. O. (1985). *The Scientific Community*. New York: Basic Books.

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

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

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

Klinker, G., J. Bentolila, S. Genetet, M. Grimes, and J. McDermott (1986). KNACK—report-driven knowledge acquisition. In: Boose, J. H. and B. R. Gaines (Eds.), *Proceedings of the First AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 23-0–23-13. Banff (November).

Marcus, S. (1987). Taking backtracking with a grain of SALT. *International Journal of Man-Machine Studies* 26(4): 383–398 (April).

Marcus, S. (Ed.) (1988). *Automating Knowledge Acquisition for Expert Systems*. New York: Kluwer.

Merton, R. K. (1973). *The Sociology of Science: Theoretical and Empirical Investigations*. Chicago: University of Chicago Press.

Michalski, R. S., and J. G. Carbonell (Eds.) (1983). *Machine Learning: An Artificial Intelligence Approach*. Palo Alto, Calif.: Tioga.

Michalski, R. S., J. G. Carbonell, and T. M. Mitchell (Eds.) (1986). *Machine Learning: An Artificial Intelligence Approach,* vol. 2. Los Altos, Calif.: Morgan Kaufmann.

Musen, M. A., L. M. Fagan, D. M. Combs, and E. H. Shortliffe (1986). Using a domain model to drive an interactive knowledge editing tool. In: J. H. Boose, and B. R. Gaines (Eds.), *Proceedings of the First AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 33-0–33-11. Banff (November).

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

Quinlan, J. R. (1987). Simplifying decision trees. *International Journal of Man-Machine Studies* 27(3): 221–234 (September).

Rappaport, A. (1987a). Cognitive primitives. In: J. H. Boose and B. R. Gaines (Eds.), *Proceedings of the Second AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 15-0–15-13. Banff (October).

Rappaport, A. (1987b). Multiple-problem subspaces in the knowledge design process. *International Journal of Man-Machine Studies* 26(4):435–452 (April).

Rappaport, A., and B. R. Gaines (1988). Integration of acquisition and performance systems. *Proceedings of the Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* pp. 25-1–25-20. Banff (November).

Rappaport, A. T., B. R. Gaines, and J. H. Boose (Eds.) (1988). *Proceedings of the First AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* St. Paul, Minn. (August).

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

Runes, D. D. (1972). *Dictionary of Philosophy*. London: Peter Owen.

Schön, D. A. (1983). *The Reflective Practitioner*. New York: Basic Books.

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

Shaw, M. L. G. (Ed.) (1981). *Recent Advances in Personal Construct Technology*. London: Academic Press.

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

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

Shaw, M. L. G., and B. R. Gaines (1986b). Techniques for knowledge acquisition and transfer. In: J. H. Boose, and B. R. Gaines (Eds.), *Proceedings of the First AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop*, pp. 39-0–39-13. Banff (November).

Shaw, M. L. G., and B. R. Gaines (1987a). An interactive knowledge elicitation techniques using personal construct technology. In: A. Kidd (Ed.), *Knowledge Elicitation for Expert Systems: A Practical Handbook*, pp. 109–136. New York: Plenum Press.

Shaw, M. L. G., and B. R. Gaines (1987b). KITTEN: Knowledge initiation and transfer tools for experts and novices. *International Journal of Man-Machine Studies* 27(3):251–280 (September).

Shaw, M. L. G., and B. R. Gaines (1988). A methodology for recognizing consensus, correspondence, conflict and contrast in a knowledge acquisition system. In: J. H. Boose and B. R. Gaines (Eds.), *Proceedings of the Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop*, pp. 30-1–30-19. Banff (November).

Shaw, M. L. G., and J. B. Woodward (1987). Validation of a knowledge support system. In: J. H. Boose and B. R. Gaines (Eds.), *Proceedings of the Second AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop*, pp. 18-0–18-15. Banff (October).

Shriver, B., and P. Wegner (1987). *Research Directions in Object-Oriented Programming*. Cambridge, Mass.: M.I.T. Press.

Slater, P. (Ed.) (1976). *Dimensions of Intrapersonal Space*, vol. 1. London: Wiley.

Stich, S. P., and R. E. Nisbett (1984). Expertise, justification and the psychology of inductive reasoning. In: T. L. Haskell (Ed.), *The Authority of Experts*, pp. 226–241. Bloomington: Indiana University Press.

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

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