KNOWLEDGE AND REQUIREMENTS ENGINEERING

Mildred L G Shaw and Brian R Gaines
Knowledge Science Institute
University of Calgary
Calgary, Alberta, Canada T2N 1N4
{mildred, gaines}@cpsc.ucalgary.ca

Abstract. An overview of knowledge engineering research, practice, theories, methodologies, and tools is presented, and parallels are drawn with analogous phenomena and activities in requirements engineering. Knowledge-based systems are distinguished from other advanced information systems by their reflective emphasis on meta-information processing about the basis of system operation. In terms of requirements elicitation this corresponds to an emphasis on maintaining an audit trail from requirements through design, implementation, use and maintenance, that supports continuing user involvement in system specification, design and evolution. Examples of knowledge elicitation methodologies and tools are given, and it is suggested that they all have some applicability in requirements elicitation for advanced information system development. It is concluded that closer collaboration between the knowledge engineering and requirements engineering communities will be mutually beneficial.

1 INTRODUCTION

This paper is concerned with the relationship between knowledge elicitation research, practice, theories, methodologies, and tools, and the parallel aspects of requirements elicitation. It focuses on the current state of the art in knowledge acquisition, and suggests potential links, sharing of experience and technology, and collaboration between the knowledge engineering and requirements engineering communities. Knowledge acquisition emerged as a distinct area of research and development in the early 1980s as a response to the need to provide scientific and engineering methodologies for the construction of expert and knowledge-based systems. In 1980 Feigenbaum (1980) coined the term "knowledge engineering", and in 1983 the book by Hayes-Roth, Waterman and Lenat (1983) on *Building Expert Systems* provided a conceptual framework for knowledge engineering and a paradigm for comparing different approaches. In 1986 the knowledge acquisition workshop series commenced and has supported the growth of an international community concerned with research, development and application of knowledge acquisition theories, methodologies and tools.

It has been recognized that the knowledge acquisition process is one of *modeling* as a basis for expert behavior, and that the "knowledge" elicited as underlying that model will often be an outcome of the modeling process rather than knowledge already existing within the expert's mental processes. It has been recognized that the transfer of that knowledge to conventional "expert system shells" is impeded by the lack of functionality and lack of knowledge/software engineering foundations for such shells. Both these shifts in perspective have brought knowledge-based system development into closer contact with issues in mainstream information systems development, notably in relation to the role of the modeling process and in relation to the architecture of advanced information systems. It has become reasonable to suggest that advances in the conceptual modeling and object-oriented architectures of databases may be providing more appropriate technologies for knowledge-based systems than those of specialist expert system shells, and that the concepts of requirements engineering substantially overlap those of knowledge engineering (Gaines, 1993).

The current state of the art in knowledge acquisition can be characterized by three major areas of advance in recent years:

1 At the conceptual and theoretical level, to view knowledge acquisition as a process of *modeling expertise* with a view to emulating and extending it.
2 At a methodological level, to provide detailed *formal modeling methodologies* supporting such processes.
3 At a technological level, to provide *computer-based tools* for knowledge acquisition supporting such modeling methodologies.

The following main sections give an overview of the state of the art in each of these three areas. Before surveying the issues in detail, however, it is interesting to discuss some broader issues which throw light on the relations between knowledge acquisition and requirements elicitation.

2 WHAT ARE KNOWLEDGE-BASED SYSTEMS?

The term "expert system" has come to be replaced in the literature by the term "knowledge-based system" as the original emphasis on transfer of human expertise to computers has been replaced by a more eclectic system development approach that makes use of all available sources. However, the presupposition remains that a "knowledge-based system" is somehow different from other advanced information systems. If so, then the scope of knowledge engineering might be somewhat more circumscribed than that of requirements and software engineering. If not so, then one has to ask why should the distinction continue to be made?

The resolution of this dilemma is to view knowledge engineering as an aspect of information system engineering—but one that places particular emphasis on the epistemological status of information that is classified as *knowledge*, that is as "justified, true belief" (Quinton, 1967) and "more than opinion, less than truth" (Runes, 1972). These definitions provide a context in which the *credibility* and *derivation* of information are significant, and need to be taken into account as overt data in their own right. That is, the knowledge-based components of an information system will tend to be those in which *meta-information* and *meta-information-processing* is required. Such systems will have components that do not only process information but also process meta-information *about* that information and its processing. A knowledge-based system may be characterized succinctly as an essentially *reflective* information system (Maes and Nardi, 1988).

This line of argument has some interesting consequences for information systems development in general. Many applications of knowledge acquisition tools and methodologies have been targeted on systems assessed by conventional performance standards with little emphasis on meta-information and reflectivity. Such applications suggest that KA tools and methodologies may be quite usefully subsumed into the general class of information system development tools and methodologies. In particular, the KA approaches that emphasize the direct involvement of experts throughout the application development process may be viewed as potential contributors to requirements elicitation and rapid system development methodologies such as Joint Application Development (August, 1991) and Participatory Design (Schuler and Namioka, 1993).

In addition, one can argue that the most important contribution of the expert system concept to date has been the emphasis of meta-information processing and reflectivity. One of the most important features in early expert systems such as MYCIN (Shortliffe, 1976) and associated development environments such as TEIRESIAS (Davis, 1982) was the capability to be reflective and answer "why" and "how" questions, and to use this capability to guide end-users in developing the system and to increase understanding of the domain (Clancey, 1987). These are significant features for any advanced information system development methodology, and lead

naturally to techniques for evolutionary design (Gilb, 1988) that continues throughout the product life cycle, and to the maintenance of requirements, design and applications experience audit trails that allow a system to present an account of its design rationale and evaluation.

In conclusion, we suggest that knowledge acquisition research be viewed in the requirements engineering community from two perspectives. First, that the theories, methodologies and tools be simply appraised in terms of their utility in system development regardless of their origins and the associated ethos of knowledge-based systems. Second, that the meta-information processing, evolutionary design, audit trail and reflective system aspects be appraised as potentially significant contributions to system design in their own right.

3 KNOWLEDGE ELICITATION AND REQUIREMENTS ELICITATION

Early research on knowledge acquisition for knowledge-based systems emphasized the acquisition of the knowledge assumed to underlie human expertise in areas such as medical diagnosis, where conventional system analysis and software engineering had failed to provide computer emulation of the expertise. This led to the "expertise transfer" paradigm in which the primary function of knowledge engineering was seen to be the transfer of a human expert's knowledge to a computer system:

> "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." (Hayes-Roth et al., 1983)

This discussion is still valuable today in characterizing the problems of eliciting knowledge from human experts. However, as numbers of knowledge-based systems were developed of increasing scope it became apparent that human experts were only one source from which knowledge was being acquired. Knowledge engineers are pragmatic in developing systems based on every available source of relevant information. It also became apparent that the status of the "knowledge" assumed to underlie human expertise was itself problematic:

> "Knowledge can be represented, but it cannot be exhaustively inventoried by statements of belief or scripts for behaving. Knowledge is a capacity to behave adaptively within an environment; it cannot be reduced to representations of behavior or the environment." (Clancey, 1989)

That is, the overt knowledge that we see in text, diagrams and computer data structures, and the invisible "knowledge" that we impute to human experts to account for their skilled behaviors are two distinct entities. It is simplistic and misleading to assume that the process that leads to the emulation of human expertise in a computer program is one of transferring knowledge—"expertise transfer" is an attractive metaphor but it leaves open the questions of what is expertise and how it may be transferred. A better metaphor might be one of modeling, that the emulation involves building a *model* of the expertise, where a "model" is according to Webster's dictionary:

> "a representation, generally in miniature, to show the construction or serve as a copy of something."

There are clearly parallels between this account of knowledge engineering and similar phenomena in requirements engineering (Greenspan, Mylopoulos and Borgida, 1994). The software engineer would like to have those who are expert about the system requirements express them clearly, concisely and completely in terms permitting implementation—that is to make their "knowledge" of the requirements overt so that the requirements may be made operational. However, the supposed knowledge may be vague, incomplete and incorrect in its initial form, and the actual knowledge that informs the system design may only arise through a process of

"elicitation" which is in fact better seen as one of modeling since what is elicited did not pre-exist but has come into being through the elicitation process.

This emergent nature of requirements has led to the criticism of the "expertise transfer" notion in knowledge engineering being paralleled by similar criticism of the "requirements capture" notion in requirements engineering:

> "There is much talk of requirements capture, as if requirements were butterflies to be caught and pinned down in a specification cabinet. In reality, of course, this is not so; requirements are often much better thought of as being elicited through such techniques as ethnomethodological study, structured interviews, dramaturgical exercises, and so on. The shift of emphasis from the requirements process as being formalising the explicit to making explicit what is merely implicit is clearly correct, but does not go far enough in many cases of organisational requirement, where the problem is one of envisioning the future rather than understanding the present." (Dobson, Blyth, Chudge and Strens, 1994)

This quotation characterizes one significant aspect of both knowledge and requirements engineering—the way in which models emerge out of a process of anticipation. Another significant common aspect is the need the 'junk' which may also emerge to be removed—that the models should be minimal in expressing only requirements, not the scaffolding of past experience on which they may be based. McDermid has characterized a good requirements specification as one which:

> "says everything which the designer needs to know in order to produce a system which satisfies the customer/users—and nothing more" (McDermid, 1994)

3.1 System Development Processes

The close relationship between knowledge engineering and requirements engineering becomes explicit when one considers in detail the system development processes involved as shown in Figure 1. At the top of the figure the informal social and psychological processes involved in knowledge and requirements *elicitation* have been separated from the formal representational manipulations involved in knowledge and software engineering. The objective of the elicitation phase at the top is to develop a precise, correct and complete system specification expressed in humanly understandable terms so that it can be checked for *validity* by the community of experts who originated it. The objective of the system engineering phase at the bottom is to implement a precise, correct and complete system specification expressed in formal terms such that the implementation can be formally *verified* as complying with the specification.

However, in both the knowledge engineering and requirements engineering communities it has become clear that the form of specification that can be validated against human requirements and the form of specification that can be verified against an implementation are rarely the same. In knowledge acquisition research it has become common to distinguish between "mediating representations" which are comprehensible to the human experts, and "formal representations" which are implementable in an operational system (Ford, Bradshaw, Adams-Webber and Agnew, 1993). An analogous vocabulary may be used for the parallel distinction between mediating specifications and formal specifications in requirements engineering. There is a major problem to ensure the *fidelity* of the transformation from humanly understandable mediating representations to implementable formal representations.

The gap between validation and verification is the most serious impediment to the development of theories, methodologies and tools for both knowledge engineering and requirements engineering. It is this gap that creates problems for the overall system evaluation in terms of user satisfaction shown on the right of Figure 1. What is most poignant about this gap is that the two communities shown at the top and bottom of the diagram are in reality often the same people in different roles, as experts or requirements specifiers in one phase of system development and as system users in another. Even if the validation of the mediating specification is carried out

thoroughly, and the verification of the system implementation is correct, there remains a central region of ambiguity where the fidelity of the transformation from a mediating representation to a formal specification is extremely difficult to check. It corresponds to the point at which both the experts' and users' comprehension of the system fails.

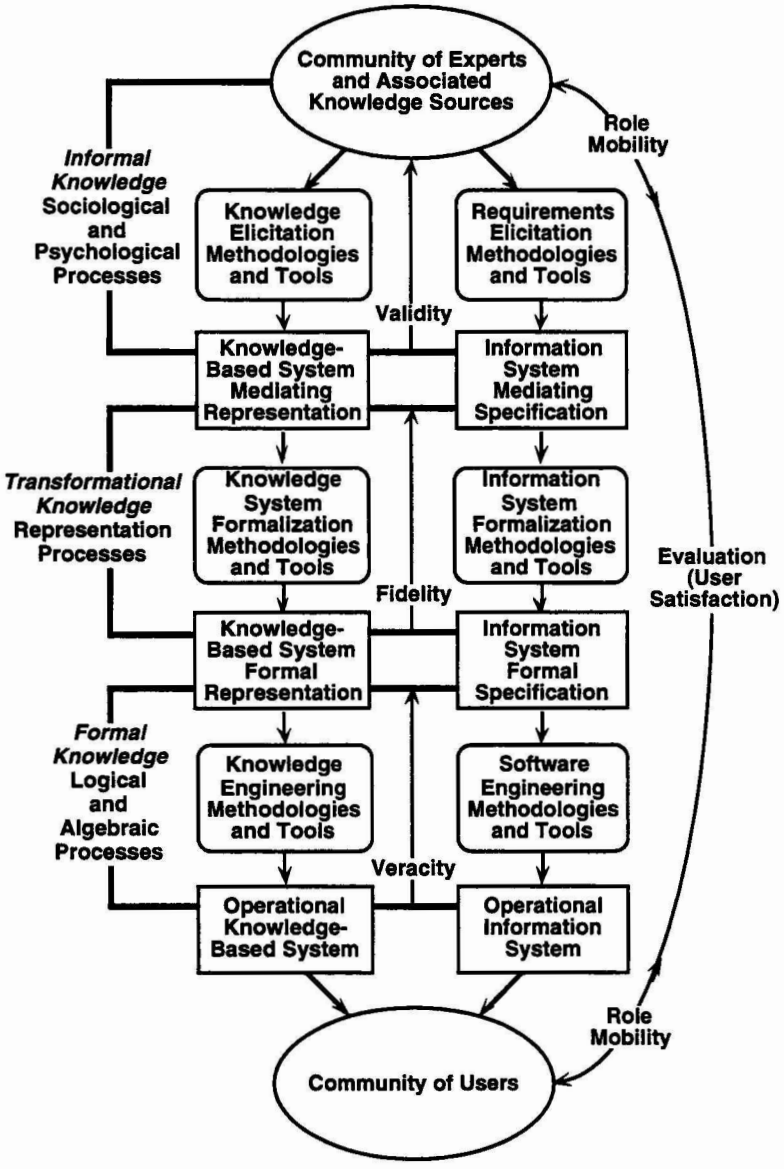


Figure 1 System development processes

One approach to overcoming the problems created by the central region of Figure 1 is to remove it altogether—to create mediating representations that are formal —to bring the expert as close as possible to the structure of an operational system. In requirements engineering this corresponds to emphasizing executable specifications and the capability to provide simulations of system functioning directly from user specifications. In knowledge acquisition this has led to an emphasis on naturalistic user interfaces to knowledge elicitation and modeling tools that attempt to make the operational knowledge structures comprehensible to the experts from whom they have been derived.

Figure 1 is relevant to different system engineering methodologies, such as the "waterfall" model, Boehm's (1988) *spiral* model, exploratory programming, evolutionary design, and so on. Different approaches emphasize different relationships and sequences in system development. However, no approach can bypass the processes shown, or legislate away the problems of bridging between the human and the formal expression of requirements.

4 KNOWLEDGE ENGINEERING AND MODELING

As discussed above, the notion that what is being done in the development of an expert system is a modeling activity has become a major theme in the literature (Ford and Bradshaw, 1993). The KADS methodology is presented as one concerned with "developing a knowledge-level model of expert reasoning" (Akkermans, Harmelen, Shreiber and Wielinga, 1993). Clancey raises the question "How do expert systems differ from conventional programs?" and answers it by:

> "expert systems contain qualitative world models...Briefly put, qualitative models describe systems in the world in terms of causal, compositional, or subtypical relationships among objects and events....Knowledge engineering is not just a new kind of programming. It is a new methodology for modeling systems so that we

can assemble, modify, and control them automatically. We are not so much programmers as engineers, scientists, and even philosophers." (Clancey, 1989)

4.1 Modeling processes in knowledge engineering

Figure 2 shows the major conceptual models that may be developed in knowledge engineering, distinguished by their sources, and indicates some of the knowledge engineering processes and skills involved (Gaines, Shaw and Woodward, 1993). This figure attempts to be comprehensive, showing knowledge sources not only in association with the expert and his or her behavior, but also knowledge derived from others, the literature and through application of laws and principles.

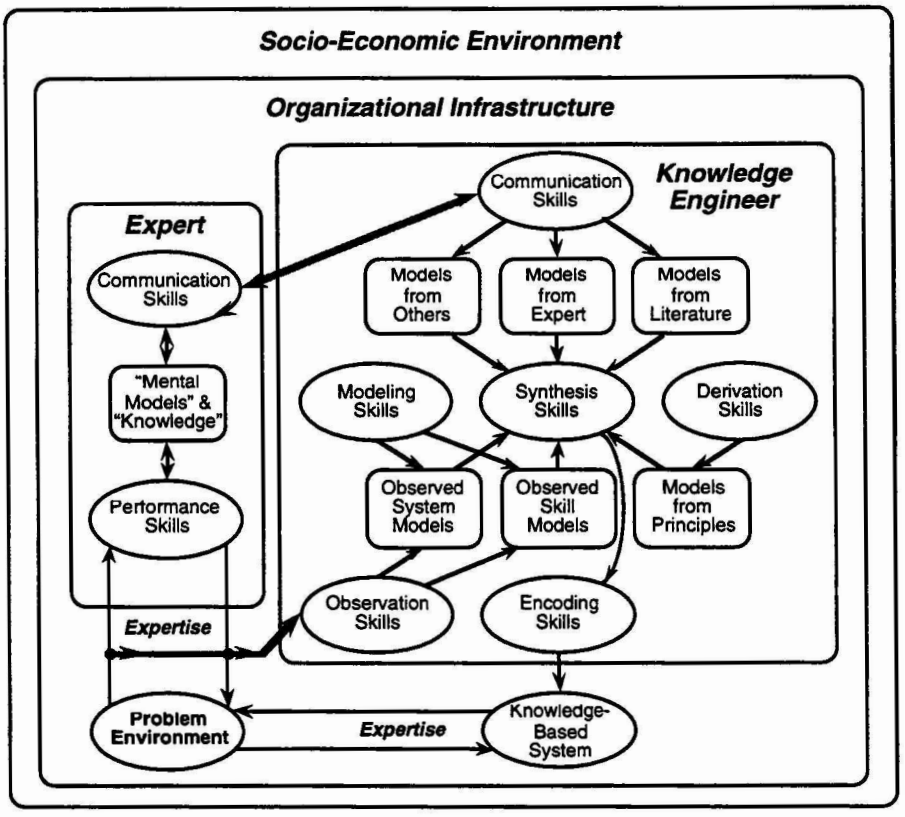


Figure 2 Knowledge sources and modeling processes in knowledge engineering

The complexity of the knowledge engineering process is very apparent in Figure 2. It is ironic that the expert may be able to function effectively with very little overt knowledge, whereas the knowledge engineer, reflecting on that expertise, becomes involved in synthesizing a model from a heterogeneous range of sources. The variety of skills demanded of the knowledge engineer seems unlikely to exist in all but a few people and suggests the need for a team approach, training programs and support tools. As emphasized already, another significant aspect of the knowledge engineering task that is apparent in Figure 2 is that the final model developed may bear little resemblance to the mental models assumed to exist within the expert. There is no reason to require that such models should be independent of the knowledge engineering process, or even exist before knowledge engineering commences.

The outer boxes in Figure 2 emphasize that neither the expert nor the knowledge engineer are completely autonomous and self-contained systems. They are each situated in an organizational infrastructure that plays a major role in providing motivation, objectives and support systems for both the expertise and its modeling. The organizational infrastructure is, in its turn, situated in a wider socio-economic environment that itself plays a major role in providing motivation,

objectives and support systems for the organization. Processes of expertise involve individuals termed 'experts' but they cannot be fully characterized by features of those experts alone. Human knowledge processes are socially situated, and their overt analysis involves modeling some aspects of the situation.

The modeling processes in Figure 2 are not all at the same level. There are fundamental differences between the observation and modeling of action, for example, and the verbal discussion of the intentions behind and logic underlying that action. It is useful to organize the modeling processes themselves within a framework that differentiates and classifies them in terms of the acquisition and analysis processes involved.

4.2 Modeling processes in software engineering

Clearly knowledge acquisition for knowledge-based systems is not uniquely characterized by its emphasis on modeling techniques. There exists in the scientific, mathematical and engineering literatures very rich frameworks encompassing the nature, function, formation and evaluation of models, including a very wide variety of techniques and tools for the development of models which have become operationalized using computers. In information technology, the notions and techniques of modeling are central to areas such as operations research and simulation—so central, in fact, that many of the key textbooks in these areas do not use 'model' as an index term since it would have so little selectivity.

If one investigates books on classical system analysis the term 'model' is far less used, often absent in both main text and index. The reason for this is significant, and can be seen best in the context of a definition of system analysis, such as that in Couger's (1973) survey of the *Evolution of Business System Analysis Technique.* He defines system analysis as concerned with two initial phases of the system development cycle:

> "Phase I—Documentation of the existing system.
> Phase II—Analysis of the system to establish requirements for an improved system (the logical design)"

These two phases, which come ahead of design and implementation, clearly satisfy the definition of a model above, in providing a representation that serves a well-defined purpose in relation to the system that is represented. Indeed that purpose may be viewed as supporting the design of an improved operational model, and it is this that probably inhibits the use of the term 'model' in the systems analysis literature for two reasons:

- There is a connotation in operations research that 'models' are operational and provide the basis for computer simulation. Hence the results of system analysis were not seen as a 'model.'
- The implementation of a system during the later stages of design and coding involves the construction of sub-systems involving structures that have little resemblance to the system being modeled. Hence the results of system implementation were not seen as a 'model.'

In recent years, as formal specification techniques have been developed, the term 'model' has come into use as a significant methodological concept:

> "In the model-based approach, specifications are explicit system models constructed out of either abstract or concrete primitives...This contrasts with the axiomatic approach where specifications were given in terms of axioms which define the relationships to each other, and thus no explicit model was formulated." (Cohen, Harwood and Jackson, 1986)

The development of formal requirements specification has been part of a three-fold move towards: proof of correctness of implementations as satisfying requirements; simulation of requirements to support system specification; automatic generation of efficient implementations directly from requirements. All of these involve introducing the operationality into system analysis that was missing in its initial history, that is a move from human interpretation of the results of system analysis to computer interpretation of those same results.

5 MODELING METHODOLOGIES

There have been two major modeling methodologies developed in knowledge acquisition research: the KADS methodology (Schreiber, Wielinga and Breuker, 1993) focusing on the derivation of the formal representation in the lower part of Figure 1; and the second, the PCP methodology (Gaines and Shaw, 1993a), focusing on the derivation of the mediating representation in the upper part of Figure 1. There are also basic modeling techniques such as Checkland's (1981) soft system methodology and Klir's (1985) epistemological hierarchy that have been used in software engineering, and encompass the range of activities in Figures 1 and 2.

5.1 KADS: a principled approach to knowledge-based system development

The KADS methodology is the outcome of a number of ESPRIT project activities centering on the University of Amsterdam but involving researchers and practitioners from many institutions, countries and disciplines. KADS is intrinsically a modeling approach with seven types of model distinguished (Schreiber et al., 1993):

5.1.1 The organizational model

An organizational model provides an analysis of the socio-organizational environment in which the knowledge-based system will have to function. It includes a description of the functions, tasks and bottlenecks in the organization. In addition it describes how the introduction of a knowledge-based system will influence the organization and the people working in it.

5.1.2 The application model

An application model defines what problem the system should solve in the organization and what the function of the system will be in this organization. In addition to the function of the knowledge-based system and the problem that it is supposed to solve, the application model specifies the external constraints that are relevant for the development of the application.

5.1.3 The task model

A task model specifies how the function of the system as specified in the application model is achieved through a number of tasks that the system will perform. Establishing this relation between function and task is not always as straightforward as it may seem. Given a goal that a system should achieve, there may be several alternative ways in which that goal can be achieved. Which alternative is appropriate in a given application depends on the characteristics of that application, on the availability of knowledge and data, and on the requirements imposed by the user or by external factors.

5.1.4 The model of cooperation

The model of cooperation contains a specification of the functionality of those sub-tasks in the task model that require a cooperative effort between the agents to whom the sub-tasks have been distributed. Some of the sub-tasks will be achieved by the system, others may be realized by the user. The result is a model of cooperative problem solving in which the user and the system together achieve a goal in a way that satisfies the various constraints posed by the task environment, the user and the state of the art of knowledge-based system technology.

5.1.5 The model of expertise

Building a model of expertise is a central activity in the process of knowledge-based system construction. It distinguishes knowledge-based system development from conventional system development. Its goal is to specify the problem solving expertise required to perform the problem-solving tasks assigned to the system. The KADS methodology focuses on expertise as the *behavior* that the system should display, and on the types of knowledge that are involved in

generating such behavior, abstracting from the details of how the reasoning is actually realized in the implementation.

5.1.6 The conceptual model

Together, the model of expertise and the model of cooperation provide a specification of the behavior of the artifact to be built. The model that results from merging these two models is similar to what is called a *conceptual model* in database development. Conceptual models are abstract descriptions of the objects and operations that a system should know about, formulated in such a way that they capture the intuitions that humans have of this behavior. The language in which conceptual models are expressed is not the formal language of computational constructs and techniques, but is the language that relates real world phenomena to the cognitive framework of the observer. In this sense conceptual models are subjective, they are relative to the cognitive vocabulary and framework of the human observer.

5.1.7 The design model

The description of the computational and representational techniques that the artifact should use to realize the specified behavior is not part of the conceptual model. These techniques are specified as separate *design decisions* in a design model. In building a design model, the knowledge engineer takes external requirements such as speed, hardware and software into account. Although there are dependencies between conceptual model specifications on the one hand and design decisions on the other hand, building a conceptual model without having to worry about system requirements makes life easier for the knowledge engineer.

The overall KADS methodology, and its supporting literature, provides a rich set of material relating to the detailed design of such models, their integration in systems, sub-methodologies and tools supporting development, and applications experience. The development of KADS and the availability of this material are major landmarks in research on knowledge acquisition for knowledge-based systems. They also represent a major area of ongoing research continuing to involve many people in many countries.

5.2 PCP: a personal construct psychology approach to knowledge acquisition

A number of methodologies and tools that have been highly influential in knowledge acquisition research have been based on personal construct psychology (PCP), Kelly's formal, constructivist model of the epistemological processes whereby people acquire expertise (Kelly, 1955). In 1980, Gaines and Shaw suggested that the tools developed by Kelly for eliciting personal models, would provide a useful development technique for expert systems (Gaines and Shaw, 1980), and performed a validation study of the elicitation of the BIAIT methodology from accountants and accounting students using computer-based repertory grid elicitation (Shaw and Gaines, 1983a). Boose in an independent parallel study reported success in a wide range of industrial expert system developments using computer elicitation of repertory grids (Boose, 1984), and since then many knowledge acquisition systems have incorporated repertory grids as a major elicitation technique (Boose and Bradshaw, 1987; Diederich, Ruhmann and May, 1987; Garg-Janardan and Salvendy, 1987; Shaw and Gaines, 1987; Ford, Cañas, Jones, Stahl, Novak and Adams-Webber, 1990).

The repertory grid is, however, only one technique for knowledge acquisition that may be derived from personal construct psychology. The formal model proposed by Kelly is highly general because of its system-theoretic derivation from the single primitive process of making dichotomous distinctions. Consideration of the recursive processes of making distinctions between distinctions leads to hierarchies of distinctions having both the generality and the complexity to encompass any model from the informality of human cognitive processes to the

formality of mathematical, axiomatic systems (Gaines and Shaw, 1984). Kelly presented his work as the foundations of a psychological system and emphasized the intensional basis of distinctions as *personal constructs* that could differ widely between individuals leading to very different personal models of the world. However, the same system is applicable to shared *social constructs* and impersonal *formal constructs* based on intensional definitions of distinctions in communal terms, or extensional definitions in concrete terms. Thus, the notion underlying personal construct psychology provide a universal foundation for modeling methodologies.

Personal construct psychology may be analyzed in a way that leads naturally to the formal derivation of KL-ONE-like knowledge representation schema (Shaw and Gaines, 1993b). Many tools have been based on personal construct psychology, including later developments of the repertory grid and visual languages for semantic networks (Bradshaw, Ford, Adams-Webber and Boose, 1993; Gaines and Shaw, 1993a).

5.3 Checkland's soft systems methodology

Checkland's (1981) soft systems methodology is a framework for system analysis that provides very powerful techniques for the analysis of systems with human and social components, and has been widely applied to difficult problem areas (Wilson, 1984). There are seven stages of system analysis in soft systems methodology as shown in Figure 3. The initial stages are concerned with system analysis and the later stages with system design. The CATWOE methodology of stage 3 is particularly interesting in its identification of the roles and expertise involved in the system definition.

Checkland's methodology prescribes six essential components of a system that must be identified at the conceptual modeling stage. The CATWOE mnemonic is a reminder to search for each of these components in the system situation and make them overt in modeling. A system is defined through a *transformation* carried out by people who are the *actors* within it. The system affects beneficially or adversely other people who are its *customers* and there is some agency with power of existence over it who is its *owner*. The system has to exist within a outside constraints forming its *environment* and the whole activity of system definition takes place within an ethos or *weltanschauung* that affects our views of it. The methodology is essentially pluralistic in emphasizing that there will generally be multiple choices for most or all of these components, and the particular choices made will result in different system models.

Stage 1: The problem situation—unstructured
Stage 2: The problem situation—expressed
Stage 3: Root definition of relevant systems—CATWOE methodology
Stage 4: Making and testing conceptual models
Stage 5: Comparing conceptual models with reality
Stage 6: Determining feasible, desirable changes
Stage 7: Action to improve the problem situation

Figure 3 Seven stages of soft systems methodology

There are natural links between personal construct psychology and soft systems analysis, and repertory grid techniques have been applied to the computerization of the CATWOE conceptual modeling process. Figure 4 shows the CATWOE analysis applied to the requirements analysis for a citizen ID system. The CATWOE analysis is essentially a triply iterated pattern of "agent-actuality" pairs, and the agents defined become the subjects for report grid elicitation of conceptual models of the actualities defined. Thus, for example, one set of grids elicited in this study was from citizens on "who to inform when I change address" (Shaw and Gaines, 1983b).

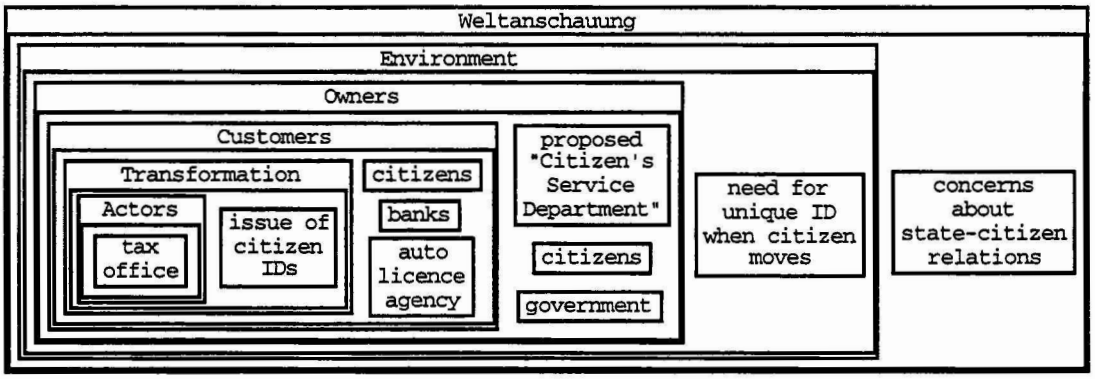


Figure 4 CATWOE analysis operationalized in a concept mapping tool

ORDIT is a requirements engineering methodology that uses Checkland's soft system analysis to implement Mumford's (1983) sociotechnical approach in which the system is viewed as a whole by placing it within the broad operational environment, with the user as an integral part of the system. Figure 5 shows the ORDIT operationalized through a concept mapping tool.

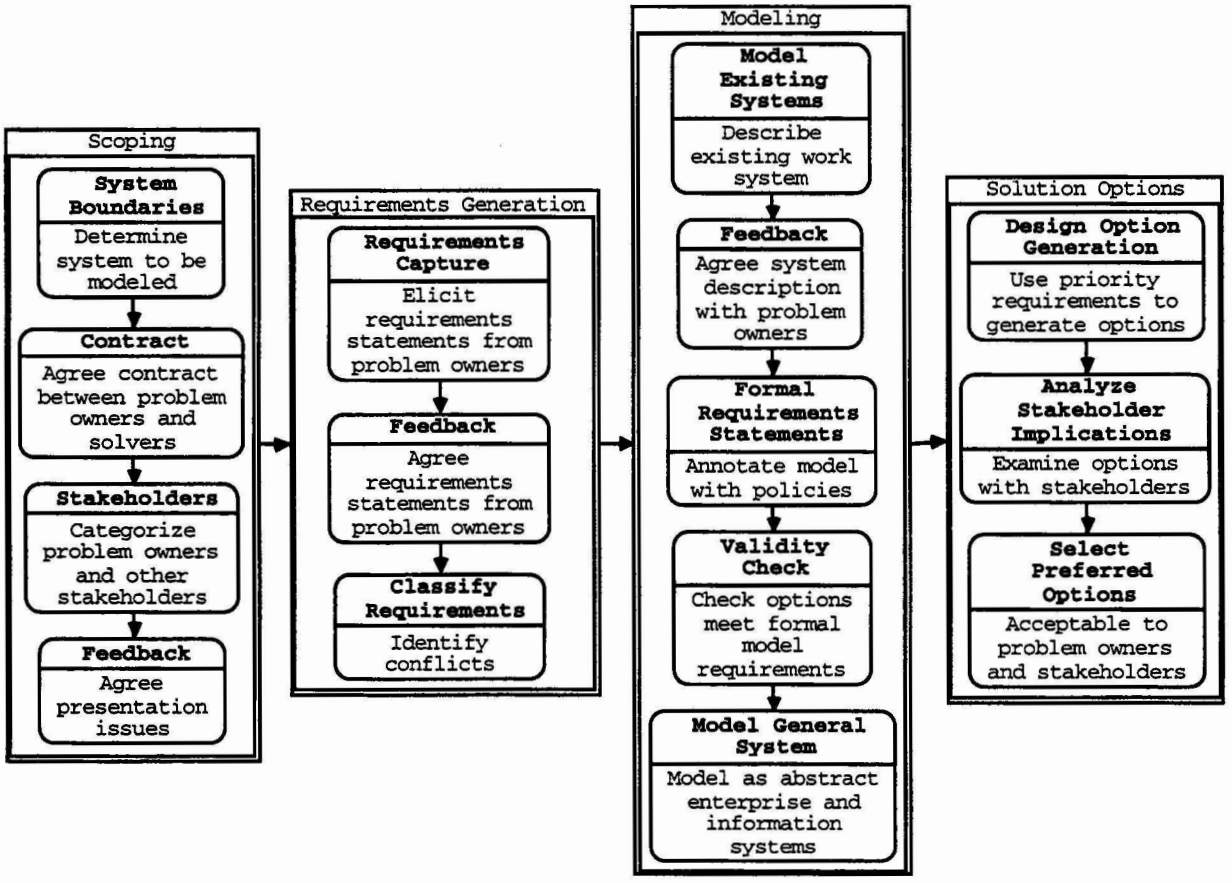


Figure 5 ORDIT requirements engineering methodology

Soft systems analysis is now widely used in requirements engineering (Dobson et al., 1994; Stamper, 1994). It is applicable to each of the seven KADS modeling areas, and provides a set of general concepts linking across areas. For example, do those who play the role of owners in one model also play the role of customers in another? Its emphasis on the context in which a system is being designed corresponds to emphasis on meta-information in knowledge-based systems.

5.4 Klir's architecture of systems problem solving

Klir architecture of systems problem solving analyses the processes involved in any modeling system in terms of an infrastructure for them that can be instantiated in different ways to encompass many different modeling schema (Klir, 1976; Klir, 1985). His basic constructs form a hierarchy of systems: a *source system* providing a descriptive terms, a *data system* providing descriptions in these terms, a *generative system* providing a regeneration of these descriptions in terms of a *structure system* providing theoretical terms, itself described through *meta-systems*, *meta-meta-systems*, etc. Figure 6 shows this modeling hierarchy operationalized in a concept mapping tool in terms of a primitive process of forming a construct by making a distinction (Gaines and Shaw, 1984). Thus, a general modeling system may itself be modeled as a process that makes distinctions in the world, gathers data in terms of those distinctions, selects from a repertoire of representations those which best generate the data, analyzes relations between the structures of such representations, and recursively repeats such analysis to generate higher levels of the hierarchy. The term *anticipation* is used for the output to capture both prediction and action. It is not necessary in general to distinguish whether the system anticipates correctly by passive prediction, or by actively changing the world to be predictable.

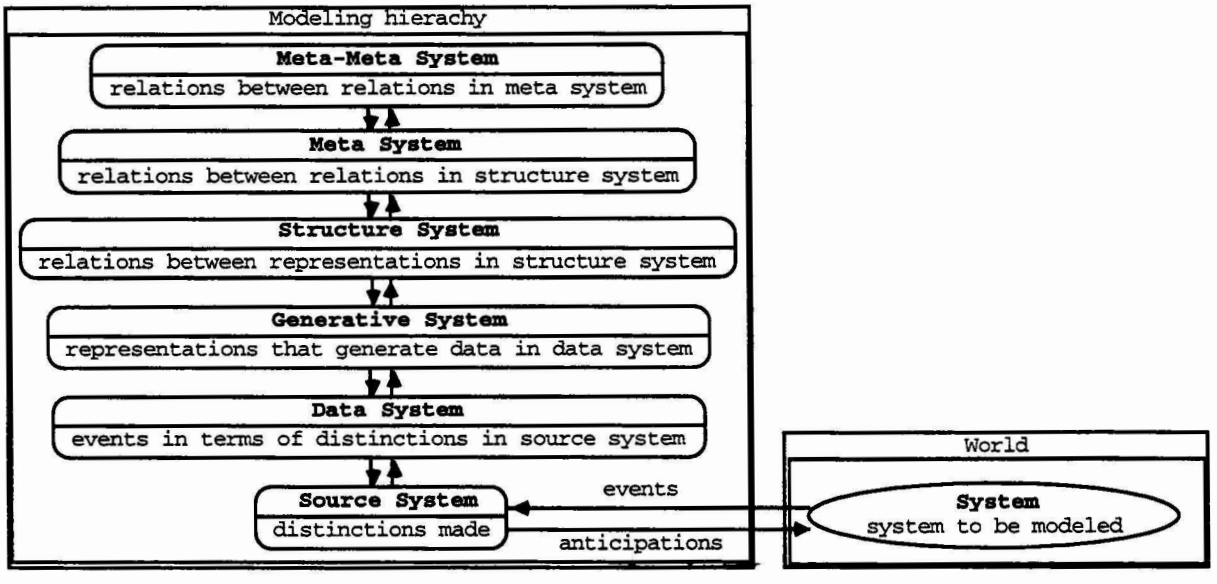


Figure 6 Klir's modeling hierarchy

The instantiation of this hierarchy with particular data description languages, model classes and measures of model complexity and model-data approximation has proven in practice to account for the wide range of mathematical modeling techniques (Gaines, 1977; Klir, 1985). It is interesting as an account of general adaptive processes since it makes clear the presuppositions necessary for modeling to take place—a *tabula rasa* cannot begin to model and some degree of 'innateness' is required. The trade-off between the amount of data needed in learning and the innately 'assumed' constraints upon the world can be investigated (Gaines, 1976), as can that with the socio-cultural filtering of data to improve its support of learning (Gaines, 1989).

Klir's modeling architecture can be viewed as a general expression of the recursive processes in distinction making at the heart of personal construct psychology, and leads to a systemic model of psychological processes (Gaines, 1989). Again, the analysis in these general terms is applicable to all of the different modeling areas of KADS, and, in general, it is apparent that there is fruitful convergence between the methodologies developed specifically for knowledge-based systems and the more general modeling methodologies of the general systems literature.

6 MODELING TOOLS

The complexity of knowledge engineering as illustrated in Figures 1 through 6, and the demands of methodologies suggest that computer support of the knowledge engineering process is essential. Computer-aided software engineering tools for knowledge-based system development have been a major theme at knowledge acquisition workshops over the years (Boose and Gaines, 1988; Boose, 1989), and may be expected to continue to be so (Gaines and Shaw, 1992).

6.1 Semi-formal elicitation and structuring through hypermedia and concept maps

Much knowledge is informal yet still valuable in a knowledge-based system. Text and pictures can encode expertise, supplementing computational knowledge. Thus, the parallel development of hypertext and hypermedia is having a substantial impact on expert system architectures and knowledge acquisition tools. Figure 7 shows some of the features of modern document processing systems that impinge on knowledge acquisition. Documents may be acquired from many sources, displayed, re-used in other documents, and linked for hypertext navigation. The text in documents may also be analyzed for associative clusters and these clusters may be grouped to indicate significant concepts (Gaines and Shaw, 1994b).

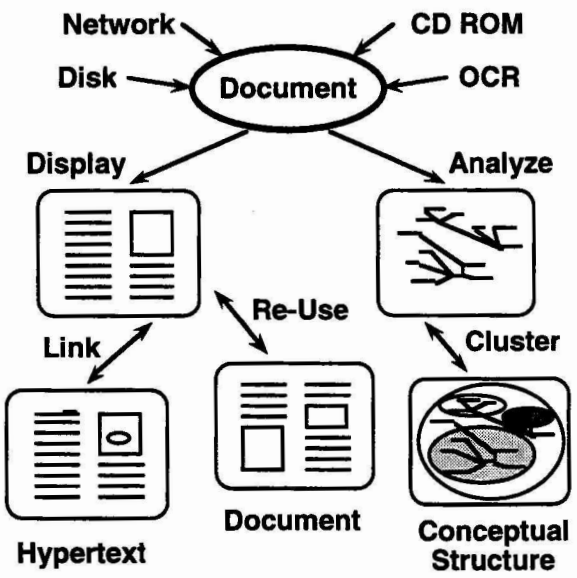


Figure 7 Hypermedia and text processing

Figure 8 shows a concept map automatically generated from a requirements document through analysis of the co-occurrence of words, a technique used in information retrieval systems (Callon, Law and Rip, 1986). The document analyzed is one on *The Technical Concept of IMS* (Tomiyama, 1992) that played a major role in the design of the Intelligent Manufacturing Systems research program. The document is treated as a set of entities which are sentences whose features are the words they contain. Rules are derived using empirical induction in which the premise is that if one word occurs in a sentence then the conclusion is that another will occur. The graph shows the links from premises to conclusions derived in this way. The tool is interactive and provides access through a popup menu associated with each word to a list of occurrences of that word in context, and to the original document.

The initial output was a digraph consisting of one major connected component and some minor ones which correspond to significant topics such as intellectual property rights that did not directly relate to the socio-technical issues. The user noted that the major component itself consisted of 3 loosely connected sub-components, and added the context boxes shown to distinguish and name these parts. The significance of these parts is that they correspond to 3 of the 5 technical work packages (TW's) of the research program. What is particularly significant is the missing work packages, TW2 concerned with product configuration management systems, and TW3 concerned with configurable production systems. These were added to the GNOSIS research program during its formation through amalgamation of interests with other potential proponents of IMS test cases. These packages link technically to the knowledge systematization activities on the left of Figure 8 but are neutral to the major issues of the IMS program on the right. Thus a text analysis system can structure the conceptual framework of requirements statements, and indicate areas that are inadequately represented (Gaines and Shaw, 1994b).

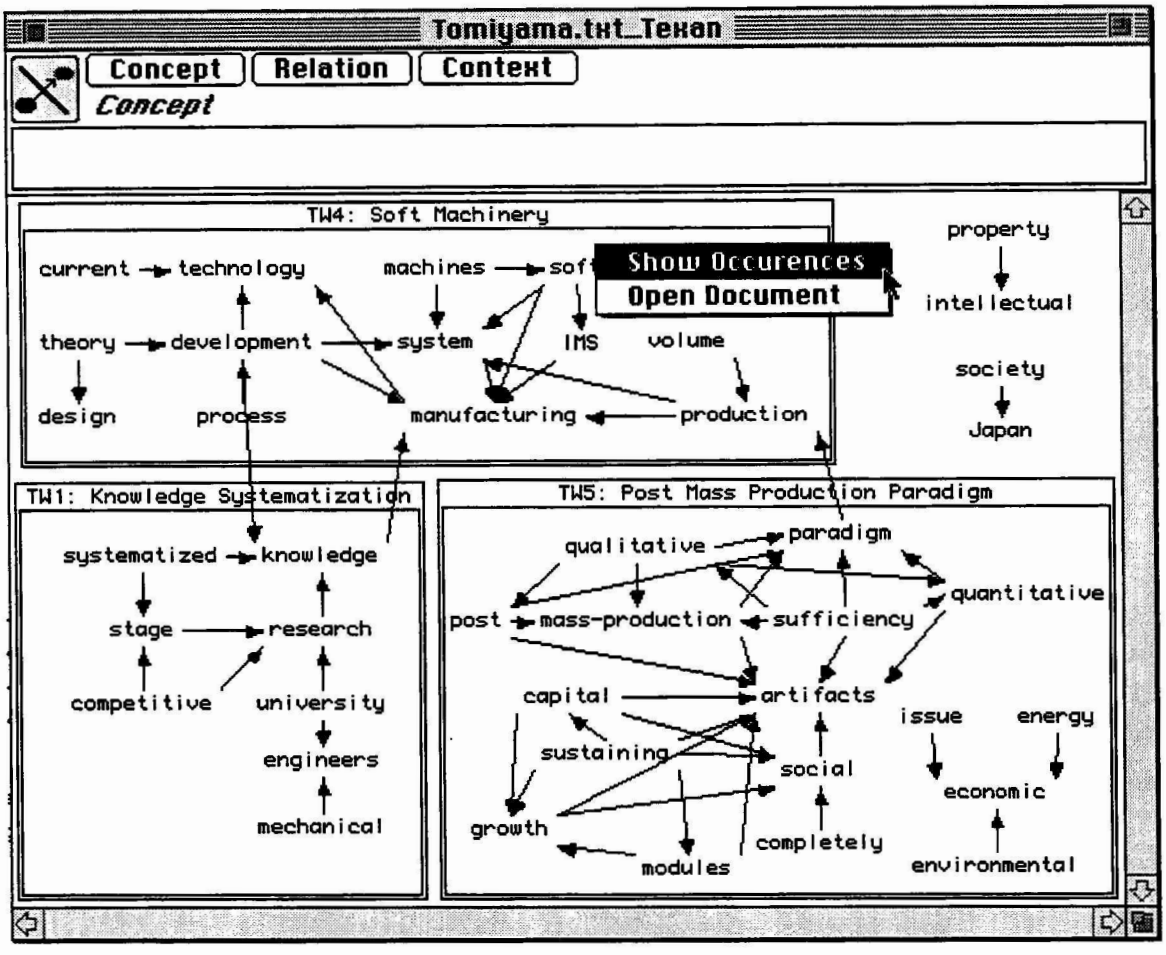


Figure 8 Concept map of a project requirements document

Hypertext-based knowledge acquisition tools have been developed for use by domain experts to enter relevant case histories directly (Jones, 1990; Rantanen, 1990). They have also been used to support the knowledge engineer in structured analyses of interview material. For example, Woodward's *Cognosys* (Woodward, 1990) supports the analysis of protocols in terms of Graesser and Clark's (Graesser and Clark, 1985) linguistically derived "general knowledge structures". Other knowledge acquisition tools such as KEATS (Motta, Eisenstadt, Pitman and West, 1988) have been built around a hypertext environment specifically designed for knowledge acquisition. There is also knowledge acquisition and linguistics research targeted on the direct transfer of knowledge expressed in text to structures of frames and rules (Gomez and Segami, 1990). Since so much knowledge is already overtly encoded in text and diagrams, in the long term this will become an essential knowledge acquisition technology. Hypertext systems have been coupled to knowledge acquisition tools to provide annotation of the distinctions made and cases described which can then be used to provide explanation facilities in the final performance system (Gaines, Linster and Shaw, 1992).

Figure 9 shows an archive of some 400MByte of multimedia data being managed through a hypermedia indexing system (Gaines and Shaw, 1994a). The system is designed for collaborative networked interaction and the map in the window at the upper left is a top level "Server Agent" that manages a particular collection of material. The concept map at the top left is currently write-disabled, and the cursor has changed to a button as the user mouses over the "Group Photo" node. Clicking at this point will display the photograph in a separate window.

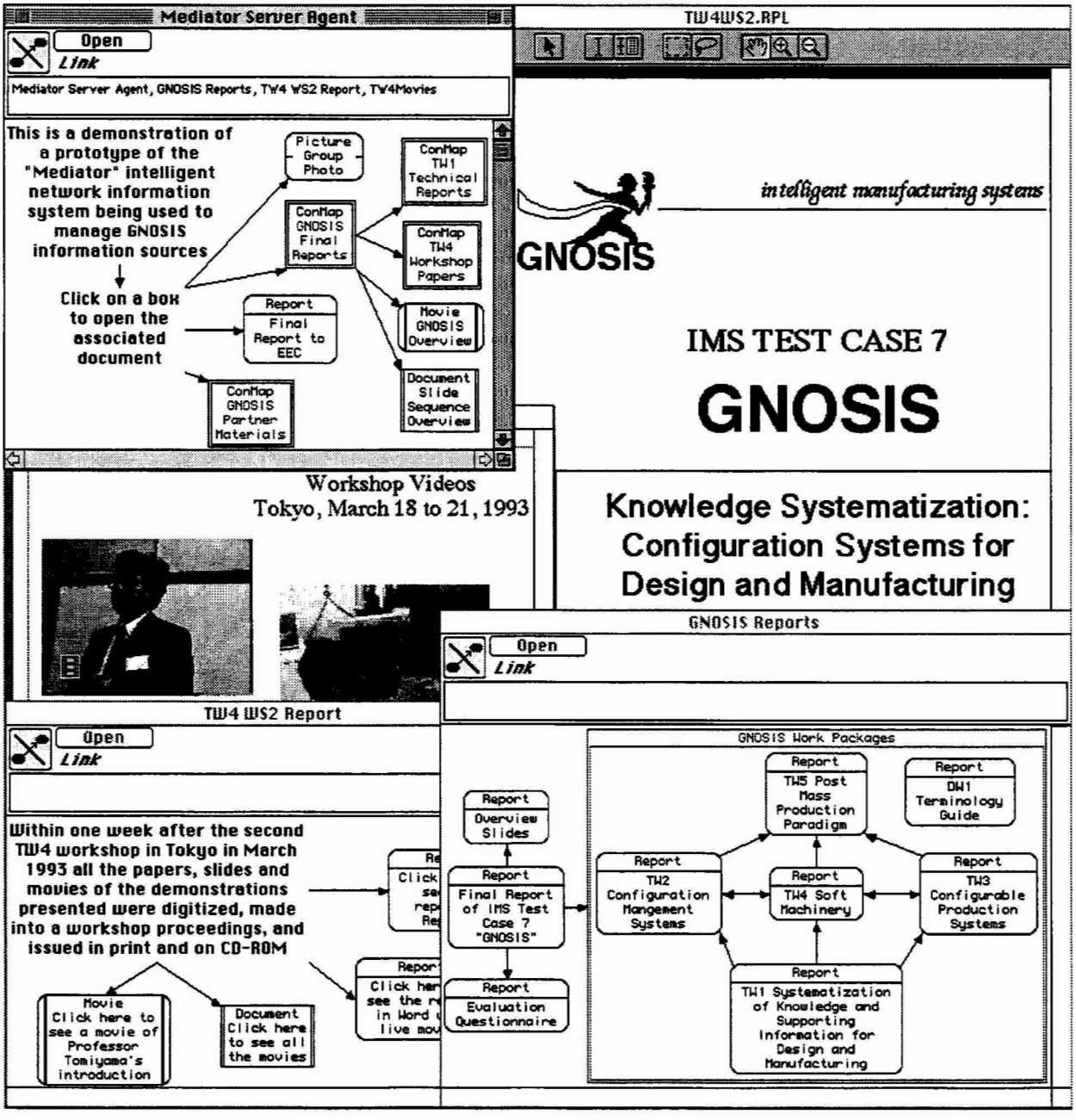


Figure 9 Accessing the GNOSIS archives through layered concept maps

The user has already clicked on the node "GNOSIS Final Reports" to open the concept map shown at the lower right. This has a node for each report, and clicking on one will open the appropriate report, in this application using Farallon's *Replica* tool. The node at the top left gives access to a series of slides on the project displayed using Persuasion's GIF *Player* tool. A similar node in the original concept map at the top left gives access to a movie on the GNOSIS project that will be opened in Apple's *MoviePlayer*.

The user has already clicked on the "TW4 Workshop Papers" node in the concept map at the top left, and opened the relevant concept map at the bottom left. She has then clicked on the node "Click here to see the report in Replica", and opened the report visible at the back on the right of Figure 9. She has also clicked on the "Click here to see all the movies" node and opened the

KWrite (Gaines and Shaw, 1993c) document visible behind the concept maps. This displays eight QuickTime movies of various demonstrations given at the Workshop, any of which can be played by double clicking on it. The network capabilities allow multiple users at geographically dispersed sites to collaborate in developing and using the archives.

The GNOSIS archives gives some feeling for how hypermedia tools may be used to index a large heterogeneous collection of requirements material. Similar developments are taking place in requirements elicitation (Christel, Wood and Stevens, 1993). The use of concept maps to support systems modeling methodologies has been shown in Figures 4 through 6.

6.2 Direct editing of knowledge in a semantic network, frame, rule, representation

Once some informal perspective on a domain has been developed and domain experts have been identified, in some domains where knowledge is already overt it may be possible to move directly to knowledge modeling. Graphic editors providing direct access to semantic network representations allowing knowledge to be encoded in frames and rules provide the most common development environment for knowledge-based systems. They are part of the application programming support environment of most expert system shells, and the widespread availability of modern graphic workstations has made it possible to provide excellent knowledge visualization environments. A wide range of knowledge acquisition tools have been developed that structure and improve the graphic editing environment, often taking advantage of domain knowledge to provide a more specific, meaningful and familiar knowledge framework to the expert. Examples are MOLE (Eshelman, Ehret, McDermott and Tan, 1987), KNACK (Klinker, Bentolila, Genetet, Grimes and McDermott, 1987), SALT (Marcus, 1987), KEATS (Motta et al., 1988) and KRS (Gaines, 1991a).

Figure 10 shows a knowledge structure being developed in KDraw, a visual language for representation language of KL-ONE knowledge structures (Gaines, 1994). The problem is one of room allocation derived from an ESPRIT project (Voß, Karbach, Drouven, Lorek and Schuckey, 1990) that was made part of Project Sisyphus. The visual language is precisely defined (Gaines, 1991b): concepts are ovals; primitive concepts are ovals with small horizontal lines inside each side; individuals are rectangles; roles are unboxed text; rules are rounded-corner boxes; and constraint expressions are rounded-corner boxes with small horizontal lines. Lines without arrows connecting primitive concepts denote that the concepts are disjoint, and connecting roles that they are inverse. The interpretation of the arrows in the editor is overloaded but well-defined by the types of the objects at their head and tail. For example: an arrow from one concept to another represents definitional subsumption, such as a "person" is an "animate" entity at the top left of Figure 10; and a concept → role → concept triple represents a definitional role with a conceptual constraint, such as an "organization" has exactly "1" (cardinality constraint) "head" who is a "head" (conceptual constraint) near the top right of Figure 10.

As with diagramming tools in software engineering, visual representation of knowledge structures with the potential for editing and enhancement is an attractive way of dealing with the results of other forms of elicitation. Semantic network editors are not so much competitors to other approaches but rather important complements to them. Many indirect knowledge acquisition tools leave the knowledge presentation and editing to the associated performance tools since these often have excellent facilities. However, in an integrated architecture it is important to incorporate editors in the knowledge acquisition tool that interact effectively with all the different forms of knowledge captured. One of the major problems to be overcome is that once the knowledge has been exported and edited in the performance system it has lost its relation to the acquisition system. Many current knowledge acquisition tools do not support long-term development and knowledge base maintenance largely because of this lack of integration.

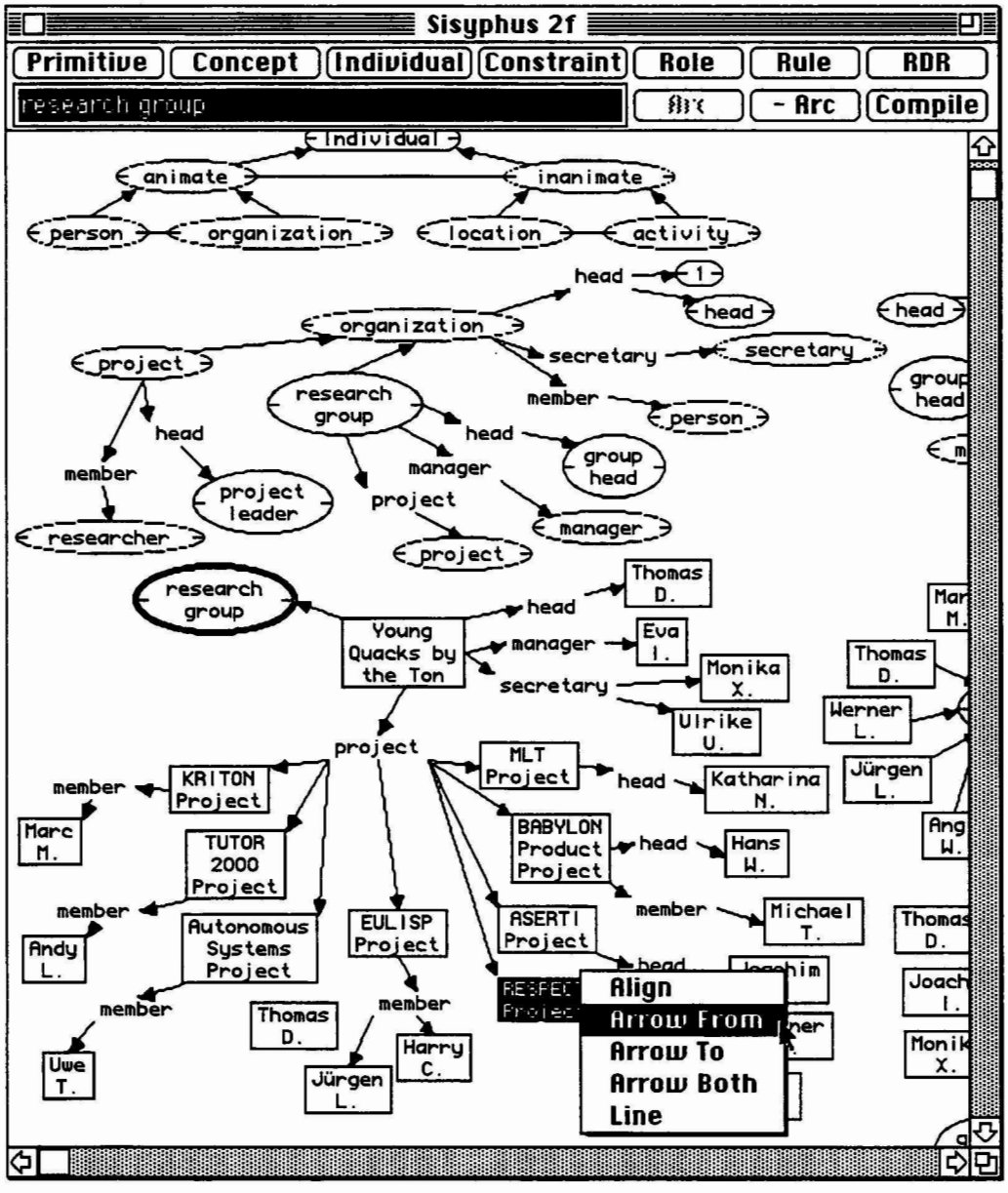


Figure 10 Visual knowledge editing as a formal semantic net in KDraw

6.3 Indirect elicitation through critical cases described in relevant attributes

The advantage of knowledge editing tools is that, if the knowledge is overtly available, they provide a fast and effective rapid prototyping environment for its direct capture. Other techniques assume that the nature of expertise is such that much knowledge is not so directly available. However, effective editing tools may be seen as essential bedrock to any integrated knowledge acquisition architecture. There will usually be significant sub-domains where direct elicitation is possible, and provides the fastest, most effective means of knowledge capture. There is always a need to present and make available for validation and editing the knowledge structures captured by indirect methods.

Repertory grids provide a technique useful for knowledge elicitation when experts cannot directly enter a knowledge structure. They prompt the expert for distinctions relevant to the problem domain and for critical cases that exhibit significant phenomena in the domain. The

prompting is done through online analysis of the data being entered leading to feedback to the expert suggesting missing distinctions and cases. This highly focused feedback aids the expert in developing his or her mental model of the domain. It also reduces the inefficiencies of duplication and the mental blocks of psychological set, supporting rapid prototyping. A wide range of knowledge acquisition tools have been developed that incorporate repertory grid elicitation and analysis as their major interface to the expert. Examples are PLANET (Shaw and Gaines, 1983a), ETS (Boose, 1984; Boose, 1986), AQUINAS (Boose and Bradshaw, 1987), KITTEN (Shaw and Gaines, 1987), and KSS0 (Gaines and Shaw, 1993b).

Figure 11 shows a screen from the repertory grid elicitation program KSS0 being used to elicit knowledge about factory layout planning. Two of the constructs elicited have been found to be highly matched and the program is showing the user the basis for the match and prompting for a new element to be entered that will not conform to it—thus attempting to ensure that false implications will not be drawn from the cases entered due to missing data.

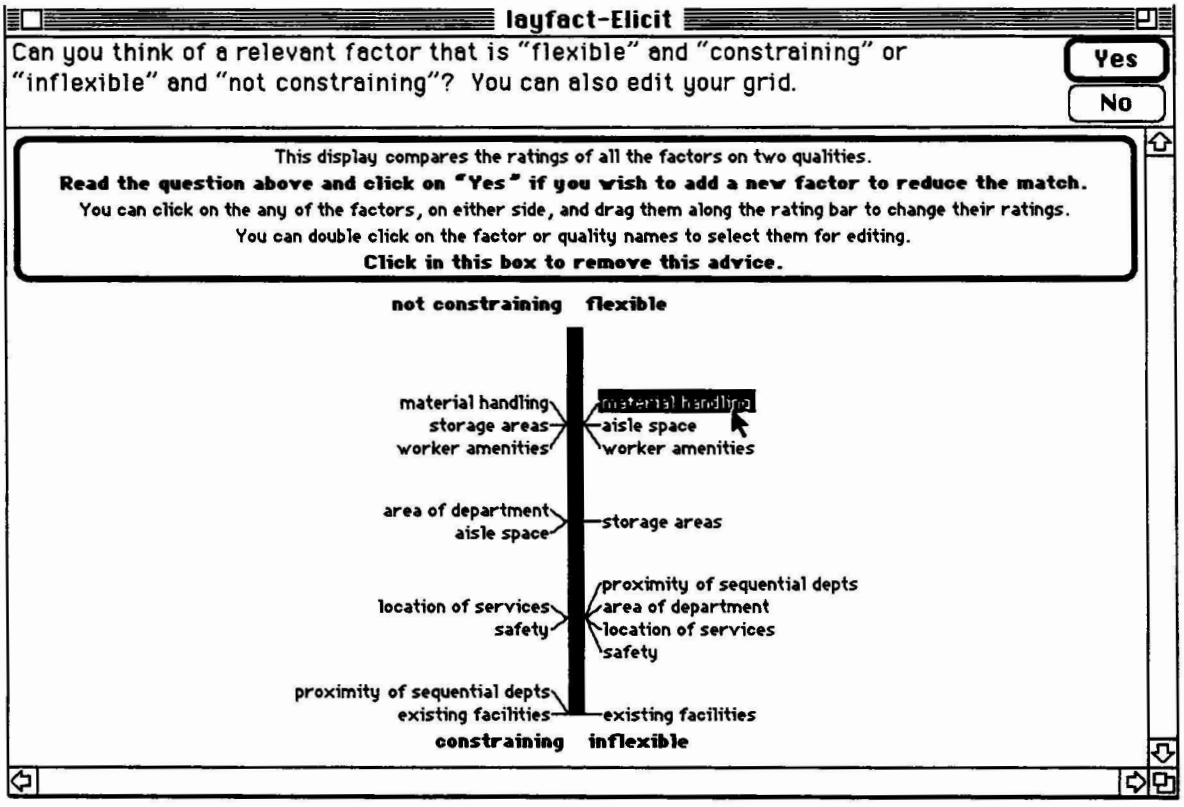


Figure 11 Construct match being used to elicit a new element

Figure 12 shows a FOCUS cluster analysis of the grid entered about layout planning. This gives an overview of correlations between both constructs and elements, allowing the domain expert to evaluate the entered data for expected consistencies.

The data from repertory grid elicited may be modeled inductively to derive rules, and exported as a set of conceptual definitions and inference rules to an expert system shell (Gaines and Shaw, 1993b). It can also be used in a similar way to define the structure of a database with the rules providing normalization or integrity constraints. Group elicitation methodologies associated with the repertory grid may be used to map the relations between the terminology and concepts used by different participants in a system design (Shaw and Gaines, 1989; Shaw and Gaines, 1993a).

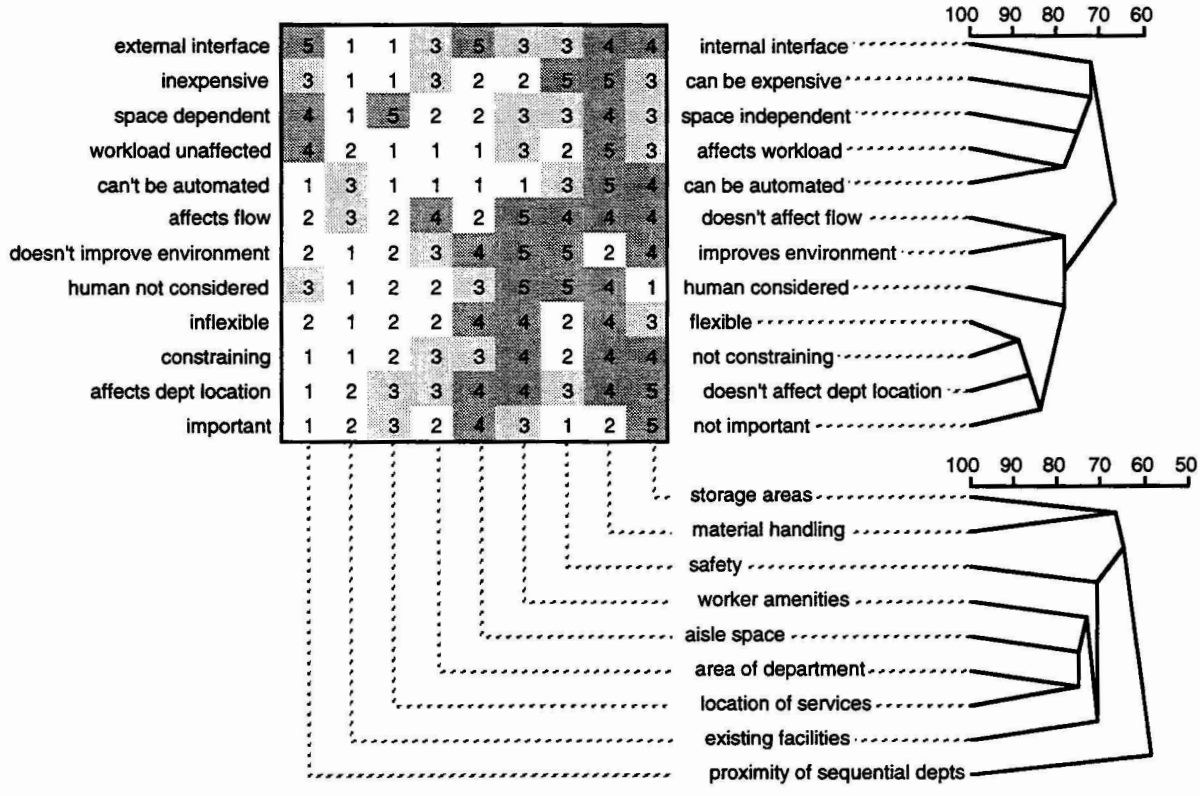


Figure 12 FOCUS cluster analysis of a repertory grid

7 CONCLUSIONS

Much of the research and practice in knowledge acquisition for knowledge-based systems during the past decade may be brought within a unified framework as the development of theories, methodologies, tools and application experience in the *modeling of expertise*. This is fortunate in enabling a very wide diversity and volume of activities to be encompassed within one conceptual framework. In particular, it captures the essence of major methodologies for knowledge acquisition such as the KADS approach to knowledge modeling and the PCP approach to human modeling processes. It is supported by general modeling methodologies such as *soft systems analysis* and the *epistemological hierarchy*, and the modeling approach is common to many software engineering methodologies.

The modeling perspective raises obvious questions as to how modeling for knowledge-based systems differs from system modeling in general, and these seem best answered in terms of fundamental definitions of knowledge itself. In any information system, if the credibility, derivation and context of information are significant, and need to be taken into account as overt data in their own right, then it is probably appropriate to take a knowledge-based approach. That is, the knowledge-based components of an information system will tend to be those in which meta-information and meta-information-processing is required. The major methodologies for knowledge acquisition all emphasize the role of meta-information, and support its acquisition and processing. The diversity of tools developed to support the knowledge engineering process may also be brought within this unified framework and characterized in terms of their sources of data and the forms of model developed from them.

From a broader perspective there is little to these tools or their associated modeling methodologies that is specific to the development of knowledge-based systems. They have direct

applicability to the development of any advanced information system. Those tools which are designed to manage and organize the large amounts of heterogeneous data gathered in the early phases of knowledge engineering are applicable to the management and organization of similar data collected in the early phases of requirements elicitation. Those tools that elicit knowledge directly from experts are applicable to the elicitation of any form of knowledge including requirements knowledge. Those tools designed to present knowledge structures in a way comprehensible to experts and users are applicable to the presentation of a wide range of computational data structures.

It would be fruitful for the knowledge engineering community to widen its horizons and analyze the role of its theories, methodologies and tools in the development of any advanced information system. It would be fruitful for the requirements engineering community to draw on the theories, methodologies and tools developed for knowledge engineering in the past decade, and re-use and extend them in more general applications. Collaborations between researchers in both communities would be mutually beneficial, and conducive to the development of scientific principles and engineering tools to support the specification, implementation and effective application of advanced information systems.

ACKNOWLEDGMENTS

Financial assistance for this work has been made available by the Natural Sciences and Engineering Research Council of Canada. We are grateful to Rudi Studer and Bill Swartout for detailed comments which improved the paper.

REFERENCES

Akkermans, H., Harmelen, F. van, Shreiber, G. and Wielinga, B. (1993). A formalisation of knowledge-level models for knowledge acquisition. **International Journal of Intelligent Systems 8**(2) 169-208.

August, J.H. (1991). **Joint Application Design: The Group Session Approach to System Design**. EngleWood Cliffs, New Jersey, Yourdon Press.

Boehm, B.W. (1988). A spiral model of software development and enhancement. **IEEE Computer 21**(5) 61-72.

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

Boose, J.H. (1986). **Expertise Transfer for Expert Systems**. Amsterdam, Elsevier.

Boose, J.H. (1989). A survey of knowledge acquisition techniques and tools. **Knowledge Acquisition 1**(1) 39-58.

Boose, J.H. and Bradshaw, J.M. (1987). 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 Gaines, B.R., Ed. (1988). **Knowledge Acquisition Tools for Expert Systems**. London, Academic Press.

Bradshaw, J.M., Ford, K.M., Adams-Webber, J.R. and Boose, J.H. (1993). Beyond the repertory grid: new approaches to constructivist knowledge acquisition tool development. **International Journal of Intelligent Systems 8**(2) 287-33.

Callon, M., Law, J. and Rip, A., Ed. (1986). **Mapping the Dynamics of Science and Technology**. Basingstoke, UK, MacMillan.

Checkland, P. (1981). **Systems Thinking, Systems Practice**. Chichester, UK, Wiley.

Christel, M.G., Wood, D.P. and Stevens, S.M. (1993). AMORE: Advanced multimedia organizer for requirements elicitation. Pittsburgh: Software Engineering Institute, Carnegie-Mellon University. CMU/SEI-93-TR-12.

Clancey, W.J. (1987). From GUIDON to NEOMYCIN and HERACLES in twenty short lessons. Lamsweerde, A. Van and Dufour, P., Ed. **Current Issues in Expert Systems**. pp.79-123. London, Academic Press.

Clancey, W.J. (1989). Viewing knowledge bases as qualitative models. **IEEE Expert 4**(2) 9-23.

Cohen, B., Harwood, W.T. and Jackson, M.I. (1986). **The Specification of Complex Systems**. Wokingham, UK, Addison-Wesley.

Couger, J.D. (1973). Evolution of business system analysis techniques. **ACM Computing Surveys 5**(3) 167-198.

Davis, R. (1982). TEIRESIAS: Applications of meta-level knowledge. Davis, R. and Lenat, D.B., Ed. **Knowledge-Based Systems in Artificial Intelligence**. pp.229-461. New York, McGraw-Hill.

Diederich, J., Ruhmann, I. and May, M. (1987). KRITON: A knowledge acquisition tool for expert systems. **International Journal of Man-Machine Studies 26**(1) 29-40.

Dobson, J.E., Blyth, A.J.C., Chudge, J. and Strens, R. (1994). The ORDIT approach to organisational requirements. Jirotka, M. and Goguen, J., Ed. **Requirements Engineering: Social and Technical Issues**. pp.87-106. London, Academic Press.

Eshelman, L., Ehret, D., McDermott, J. and Tan, M. (1987). MOLE: A tenacious knowledge acquisition tool. **International Journal of Man-Machine Studies 26**(1) 41-54.

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

Ford, K.M. and Bradshaw, J.M., Ed. (1993). **Knowledge Acquisition as Modeling**. New York, Wiley.

Ford, K.M., Bradshaw, J.M., Adams-Webber, J.R. and Agnew, N.M. (1993). Knowledge acquisition as a constructive modeling activity. **International Journal of Intelligent Systems 8**(1) 9-32.

Ford, K.M., Cañas, A., Jones, J., Stahl, H., Novak, J. and Adams-Webber, J. (1990). ICONKAT: an integrated constructivist knowledge acquisition tool. **Knowledge Acquisition 3**(2) 215-236.

Gaines, B.R. (1976). Behaviour/structure transformations under uncertainty. **International Journal Man-Machine Studies 8**(3) 337-365.

Gaines, B.R. (1977). System identification, approximation and complexity. **International Journal of General Systems 2**(3) 241-258.

Gaines, B.R. (1989). Social and cognitive processes in knowledge acquisition. **Knowledge Acquisition 1**(1) 251-280.

Gaines, B.R. (1991a). Empirical investigations of knowledge representation servers: Design issues and applications experience with KRS. **ACM SIGART Bulletin 2**(3) 45-56.

Gaines, B.R. (1991b). An interactive visual language for term subsumption visual languages. **IJCAI'91: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence**. pp.817-823. San Mateo, California, Morgan Kaufmann.

Gaines, B.R. (1993). Modeling and extending expertise. Aussenac, N., Boy, G., Gaines, B., Linster, M., Ganascia, J.G. and Kodratoff, Y., Ed. **EKAW'93: Knowledge Acquisition for Knowledge-Based Systems**. pp.1-22. Berlin, Springer.

Gaines, B.R. (1994). A situated classification solution of a resource allocation task represented in a visual language. **International Journal Human-Computer Studies 40**(2) 243-271.

Gaines, B.R., Linster, M. and Shaw, M.L.G. (1992). An integrated knowledge support system. **Proceedings of FGCS'92: International Conference on Fifth Generation Computer Systems**. pp.1157-1164. Tokyo, ICOT.

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

Gaines, B.R. and Shaw, M.L.G. (1984). Hierarchies of distinctions as generators of system theories. Smith, A.W., Ed. **Proceedings of the Society for General Systems Research International Conference**. pp.559-566. Louisville, Kentucky, Society for General Systems Research.

Gaines, B.R. and Shaw, M.L.G. (1992). Integrated knowledge acquisition architectures. **Journal for Intelligent Information Systems 1**(1) 9-34.

Gaines, B.R. and Shaw, M.L.G. (1993a). Basing knowledge acquisition tools in personal construct psychology. **Knowledge Engineering Review 8**(1) 49-85.

Gaines, B.R. and Shaw, M.L.G. (1993b). Eliciting knowledge and transferring it effectively to a knowledge-based systems. **IEEE Transactions on Knowledge and Data Engineering 5**(1) 4-14.

Gaines, B.R. and Shaw, M.L.G. (1993c). Open architecture multimedia documents. **Proceedings of ACM Multimedia 93**. pp.137-146.

Gaines, B.R. and Shaw, M.L.G. (1994a). Concept maps indexing multimedia knowledge bases. **AAAI-94 Workshop: Indexing and Reuse in Multimedia Systems**. pp.36-45. Menlo Park, California, AAAI.

Gaines, B.R. and Shaw, M.L.G. (1994b). Using knowledge acquisition and representation tools to support scientific communities. **AAAI'94: Proceedings of the Twelfth National Conference on Artificial Intelligence**. pp.707-714. Menlo Park, California, AAAI Press/MIT Press.

Gaines, B.R., Shaw, M.L.G. and Woodward, J.B. (1993). Modeling as a framework for knowledge acquisition methodologies and tools. **International Journal of Intelligent Systems 8**(2) 155-168.

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

Gilb, T. (1988). **Principles of Software Engineering Management**. Wokingham, UK, Addison-Wesley.

Gomez, F. and Segami, C. (1990). Knowledge acquisition from natural language for expert systems based on classification problem-solving methods. **Knowledge Acquisition 2**(2) 107-128.

Graesser, A.C. and Clark, L.F. (1985). **Structures and Procedures of Implicit Knowledge**. New Jersey, Ablex.

Greenspan, S., Mylopoulos, J. and Borgida, A. (1994). On formal requirements modeling languages: RML revisited. **Proceedings 16th International Conference on Software Engineering**. pp.135-148. New York, IEEE Press.

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

Jones, W.P. (1990). Bringing corporate knowledge into focus with CAMEO. **Knowledge Acquisition 2**(3) 207-239.

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

Klinker, G., Bentolila, J., Genetet, S., Grimes, M. and McDermott, J. (1987). KNACK—report-driven knowledge acquisition. **International Journal of Man-Machine Studies 26**(1) 65-79.

Klir, G.J. (1976). Identification of generative structures in empirical data. **International Journal of General Systems, 3** 89-104.

Klir, G.J. (1985). **Architecture of Systems Problem Solving**. New York, Plenum Press.

Maes, P. and Nardi, D., Ed. (1988). **Meta-Level Architectures and Reflection**. Amsterdam, North-Holland.

Marcus, S. (1987). Taking backtracking with a grain of SALT. **International Journal of Man-Machine Studies 26**(4) 393-398.

McDermid, J.A. (1994). Requirements analysis: Orthodox, fundamentalism and heresy. Jirotka, M. and Goguen, J., Ed. **Requirements Engineering: Social and Technical Issues**. pp.17-40. London, Academic Press.

Motta, E., Eisenstadt, M., Pitman, K. and West, M. (1988). Support for knowledge acquisition in the knowledge engineer's assistant (KEATS). **Expert Systems 5**(1) 6-28.

Mumford, E. (1983). **Designing Human Systems**. Manchester, UK, Manchester Business School Publications.

Quinton, A. (1967). Knowledge and belief. Edwards, P., Ed. **The Encyclopedia of Philosophy: Volume 4**. pp.345-352. New York, Macmillan.

Rantanen, J.A. (1990). Hypermedia in knowledge acquisition and specification of user interface for KBS: an approach and a case study. **Knowledge Acquisition 2**(3) 259-278.

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

Schreiber, A.Th., Wielinga, B.J. and Breuker, J.A., Ed. (1993). **KADS: A Principled Approach to Knowledge-based System Development**. London, Academic Press.

Schuler, D. and Namioka, A., Ed. (1993). **Participatory Design**. Hillsdale, New Jersey, Erlbaum.

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

Shaw, M.L.G. and Gaines, B.R. (1983b). Eliciting the real problem. Wedde, H., Ed. **Adequate Modeling of Systems**. pp.100-111. Berlin, Springer.

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

Shaw, M.L.G. and Gaines, B.R. (1989). A methodology for recognizing conflict, correspondence, consensus and contrast in a knowledge acquisition system. **Knowledge Acquisition 1**(4) 341-363.

Shaw, M.L.G. and Gaines, B.R. (1993a). Group knowledge elicitation over networks. Bramer, M.A. and Macintosh, A.L., Ed. **Research and Development in Expert Systems X. Proceedings of British Computer Society Expert Systems Conference**. pp.43-62. Cambridge, UK, Cambridge University Press.

Shaw, M.L.G. and Gaines, B.R. (1993b). Personal construct psychology foundations for knowledge acquisition and representation. Aussenac, N., Boy, G., Gaines, B., Linster, M., Ganascia, J.G. and Kodratoff, Y., Ed. **Proceedings of EKAW-93: Seventh European Workshop on Knowledge Acquisition for Knowledge-Based Systems**. pp.256-276.

Shortliffe, E.H. (1976). **Computer-Based Medical Consultations: MYCIN**. New York, Elsevier.

Stamper, T. (1994). Social norms in requirements analysis—an outline of MEASUR. Jirotka, M. and Goguen, J., Ed. **Requirements Engineering: Social and Technical Issues**. pp.107-139. London, Academic Press.

Tomiyama, T. (1992). The technical concept of IMS. RACE Discussion Paper, No. RA-DP2, Research into Artifacts, Center for Engineering, The University of Tokyo.

Voß, A., Karbach, W., Drouven, U., Lorek, D. and Schuckey, R. (1990). Operationalization of a synthetic problem. **ESPRIT Basic Research Project P3178 REFLECT Task I.2.1 Report**. Bonn, Germany, GMD.

Wilson, B. (1984). **Systems: Concepts, Methodologies, and Applications**. Chichester, UK, Wiley.

Woodward, B. (1990). Knowledge engineering at the front-end: defining the domain. **Knowledge Acquisition 2**(1) 73-94.