Requirements acquisition

by Mildred L.G. Shaw and Brian R. Gaines

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 provided, 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 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. During this period Feigenbaum [1] coined the term 'knowledge engineering', and Hayes-Roth *et al.* [2] 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 recognised that the knowledge acquisition process is one of *modelling* as a basis for expert behaviour, and that the 'knowledge' elicited as underlying that model will often be an outcome of the modelling process, rather than knowledge already existing within the expert's mental processes. It has been recognised 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 modelling process and in relation to the architecture of advanced information systems. It has become reasonable to suggest that advances in the conceptual modelling 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 [3].

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

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

The following provides 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 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, then we have 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*, i.e. as 'justified, true belief' [4] and 'more than opinion, less than truth' [5]. 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; i.e. 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.

One of the most important features in early expert systems such as MYCIN [6] and associated development environments such as TEIRESIAS [7] 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 [8]. These are significant features for any advanced information system development methodology, and lead naturally to techniques for evolutionary design [9] that continues throughout the product life cycle, and to the maintenance of requirements, design and applications audit trails that allow a system to present an account of its design rationale and evaluation.

From the meta-information processing perspective, a knowledge-based system may be characterized as an essentially *reflective* information system [10]. This suggests that the extent of embedded meta-information processing is what differentiates conventional and knowledge-based systems. For example, requirements engineering may use qualitative simulation as a design technique, whereas knowledge engineering will use it as an embedded system component.

In practice, many applications of knowledge acquisition (KA) 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 emphasise 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 [11] and Participatory Design [12].

In conclusion, we suggest that knowledge acquisition research is viewed in the requirements engineering community from two perspectives. First, the theories, methodologies and tools should be simply appraised in terms of their utility in system development, regardless of their origins and the associated ethos of knowledge-based systems. Secondly, the meta-information processing, audit trail and reflective system aspects should 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 emphasised 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.' [2]

This discussion is still valuable in characterising the problems of eliciting knowledge from human experts. However, as more knowledge-based systems of increasing scope were being developed, 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.' [13]

This means that the overt knowledge in text, diagrams and computer data structures and the invisible 'knowledge' that we impute to human experts to account for their skilled behaviours 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 modelling, that the emulation involves building a *model* of the expertise, where, according to Webster's dictionary, a 'model' is

'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 [14]. 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; i.e. 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 modelling because 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' 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'. [15]

This quotation characterises 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 for the 'junk' that 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 characterised 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'. [16]

3.1 System development processes

The close relationship between knowledge engineering and requirements engineering becomes explicit when you consider in detail the system development processes involved, as shown in Fig. 1. At the top of the Fig. 1 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 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 [17]. An analogous vocabulary may be used for the parallel distinction between mediating specifications and formal specifications in requirements engineering. It is a major problem to ensure the *fidelity* of the transformation from 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 Fig. 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; experts or requirements specifiers in one phase of system development and 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.

One approach to overcoming the problems created by the central region of Fig. 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 emphasising 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 modelling tools that attempt to make the operational knowledge structures comprehensible to the experts from whom they have been derived.

Fig. 1 encompasses many different system engineering methodologies, such as the various waterfall models, Boehm's [18] *spiral* models, *exploratory programming*, *evolutionary design*, etc. 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 modelling

As discussed above, the notion that what is being done in the development of an expert system is a modelling activity has become a major theme [19]. The KADS method-

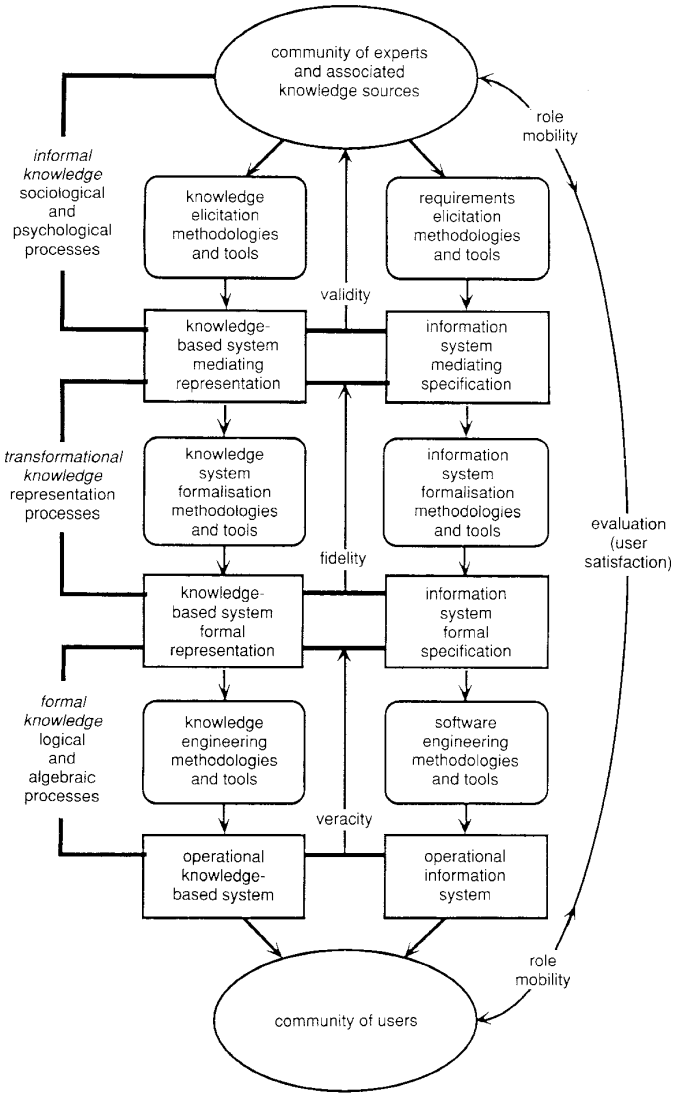

Fig. 1 System development processes

ology is presented as one concerned with 'developing a knowledge-level model of expert reasoning' [20]. Clancey [13] raises the question 'How do expert systems differ from conventional programs?' and answers it with

'*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.*' [13]

4.1 Modelling processes in knowledge engineering

Fig. 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 [21]. Fig. 2 attempts to be comprehensive, showing knowledge sources not only in association with the expert and his or her behaviour, but also knowledge derived from others, the literature and through application of laws and principles.

The complexity of the knowledge engineering process is very apparent in Fig. 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 synthesising 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 emphasised already, another significant aspect of the knowledge engineering task apparent in Fig. 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 Fig. 2 emphasise that neither the expert nor the knowledge engineer are completely autonomous and self-contained systems. They are each situated in an organisational infrastructure that plays a major role

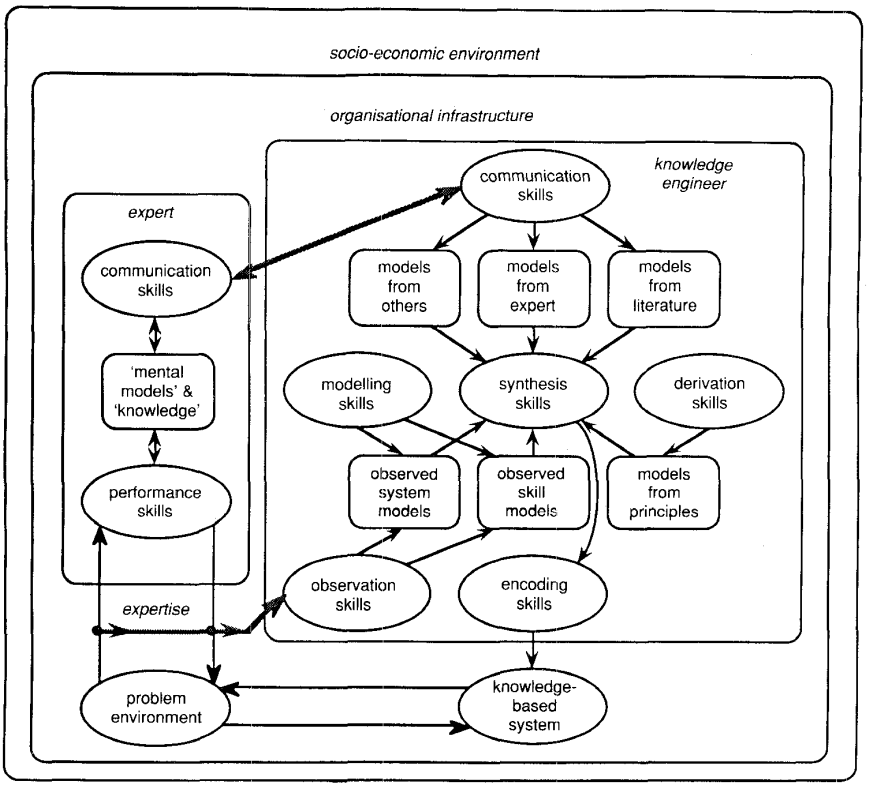

Fig. 2 Knowledge sources and modelling processes in knowledge engineering

in providing motivation, objectives and support systems for both the expertise and its modelling. The organisational 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 organisation. Processes of expertise involve individuals called 'experts', but they cannot be fully characterised by features of those experts alone. Human knowledge processes are socially situated, and their overt analysis involves modelling some aspects of the situation.

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

4.2 *Modelling processes in software engineering*

Clearly, knowledge acquisition for knowledge-based systems is not uniquely characterised by its emphasis on modelling techniques. In the literature there are very rich frameworks encompassing the nature, function, formation and evaluation of models, including a very wide variety of techniques and tools for model development which have become operationalised using computers. In information technology, the notions and techniques of modelling 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 because it would have so little selectivity.

If we investigate textbooks on classical system analysis the term 'model' is used far less, often absent in both main text and index. The reason for this is significant and can best be seen in the context of a definition of system analysis, such as that in Couger's [22] survey. 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 above definition of a model, 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 with structures that have little resemblance to the system being modelled; 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.' [23]

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; and 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; a move from human interpretation of the results of system analysis to computer interpretation of those same results.

5 Modelling methodologies

There have been two major modelling methodologies developed in knowledge acquisition research: the KADS methodology [24] focusing on the derivation of the formal representation in the lower part of Fig. 1; and the PCP methodology [25], focusing on the derivation of the mediating representation in the upper part of Fig. 1. There are also basic modelling techniques such as Checkland's [26] soft system methodology and Klir's [27] epistemological hierarchy that have been used in software engineering, and encompass the range of activities in Figs. 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, centring on the University of Amsterdam, but involving researchers and practitioners from many institutions, countries and disciplines. KADS is intrinsically a modelling approach with seven types of model [24].

5.1.1 Organisational model: provides an analysis of the socio-organisational environment in which the knowledge-based system will have to function. It includes a description of the functions, tasks and bottle-necks in the organisation. In addition it describes how the introduction of a knowledge-based system will influence the organisation and the people working in it.

5.1.2 Application model: defines what problem the system should solve in the organisation and what the function of the system will be in this organisation. 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 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 Model of co-operation: contains a specification of the functionality of those sub-tasks in the task model that require a co-operative 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 realised by the user. The result is a model of co-operative 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 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 *behaviour* that the system should display, and focuses on the types of knowledge that are involved in generating such behaviour, abstracting from the details of how the reasoning is actually realised in the implementation.

5.1.6 Conceptual model: together, the model of expertise and the model of co-operation provide a specification of the behaviour of the artefact 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 behaviour. The language in which conceptual models are expressed is not the formal language of computational constructs and techniques, but 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 Design model: the description of the computational and representational techniques that the artefact should use to realise the specified behaviour 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 into account external requirements such as speed, hardware and software. 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 is 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 involving many people in many countries.

5.2 PCP approach to knowledge acquisition

Several methodologies and tools that have been highly influential in knowledge acquisition research have been based on Kelly's personal construct psychology (PCP) [28]; a formal, constructivist model of the epistemological processes whereby people acquire expertise. In 1980, we suggested that the tools developed by Kelly for eliciting personal models would provide a useful development technique for expert systems [29]. We performed a validation study of the elicitation of the BIAIT methodology from accountants and accounting students using computer-based repertory grid elicitation [30]. In an independent parallel study, Boose reported success in a wide range of industrial expert system developments using computer elicitation of repertory grids [31], and since then many knowledge acquisition systems have incorporated repertory grids as a major elicitation technique [32–36].

The repertory grid is, however, only one technique for knowledge acquisition that may be derived from PCP. The formal model proposed by Kelly is extremely 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 with both the generality and the complexity to encompass any model, from the informality of human cognitive processes to the formality of mathematical, axiomatic systems [37]. Kelly presented his work as the foundation of a psychological system and emphasised the intentional 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 intentional definitions of distinctions in communal terms or extentional definitions in concrete terms. Thus, the notion underlying personal construct psychology provides a universal foundation for modelling methodologies.

PCP may be analysed in a way that leads naturally to the formal derivation of KL-ONE-like knowledge representation schema [38]. Many tools have been based on personal construct psychology, including later developments of the repertory grid and visual languages for semantic networks [25, 39].

Stage 1: The problem situation—unstructured
Stage 2: The problem situation—expressed
Stage 3: Root definition of relevant systems—CATWOE
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

Fig. 3 Seven stages of soft systems methodology

5.3 Checkland's soft systems methodology

Checkland's [26] 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 it has been widely applied to difficult problem areas [40]. There are seven stages of system analysis in soft systems methodology, as shown in Fig. 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 modelling stage. The CATWOE mnemonic is a reminder to search for each of these components in the system situation and make them overt in modelling. 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 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 emphasising 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.

There are natural links between personal construct psychology and soft systems analysis, and repertory grid techniques have been applied to the computerisation of the CATWOE conceptual modeling process. Fig. 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

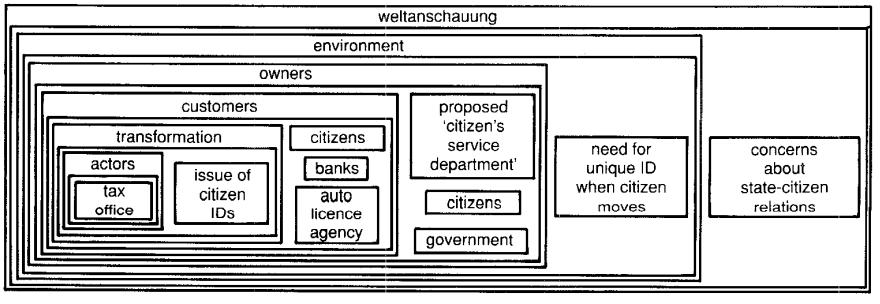

Fig. 4 CATWOE analysis operationalised in a concept mapping tool

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' [41].

Soft systems analysis is applicable to each of the seven KADS modelling areas and provides a set of general concepts linking across areas. For example, do those who play the role of owners in one model play the role of customers in another? Its emphasis on the context in which a system is being designed corresponds to an emphasis on meta-information in knowledge-based systems.

ORDIT is a requirements engineering methodology that uses soft systems analysis to implement Mumford's [42] 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 [15]. The aim of the ORDIT Project is to develop a methodology that will enable designers to reason about organisational goals, policies and structures, and the work roles for intended end users, and hence identify and support the requirements for the system. ORDIT techniques provide a rich environment for the representation of the organisational structure and the relationships between the roles of agents, information flows, and resource management [43]. Responsibilities and relationships are modelled rather than activities, and information is modelled from the point of view of the contracts involved. The process includes the identification of the requirements owners, together with their positions and roles in the organisation, and the identification of the user community and others affected by the system, together with their roles and responsibilities in the organisation [44, 45]. Fig. 5 shows the ORDIT methodology managed through a concept mapping tool.

5.4 *Klir's architecture of systems problem solving*

Klir's [27, 46] architecture of systems problem solving analyses the processes involved in any modelling system in terms of an infrastructure that can be instantiated in different ways to encompass many different modelling schema. His basic constructs form a hierarchy of systems: a *source system* providing 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.

Fig. 6 shows this modelling hierarchy operationalised in a concept mapping tool, in terms of a primitive process of forming a construct by making a distinction [37]. Thus, a general modelling system may itself be modelled 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; analyses 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.

The instantiation of this hierarchy with particular data description languages, model classes and measures of

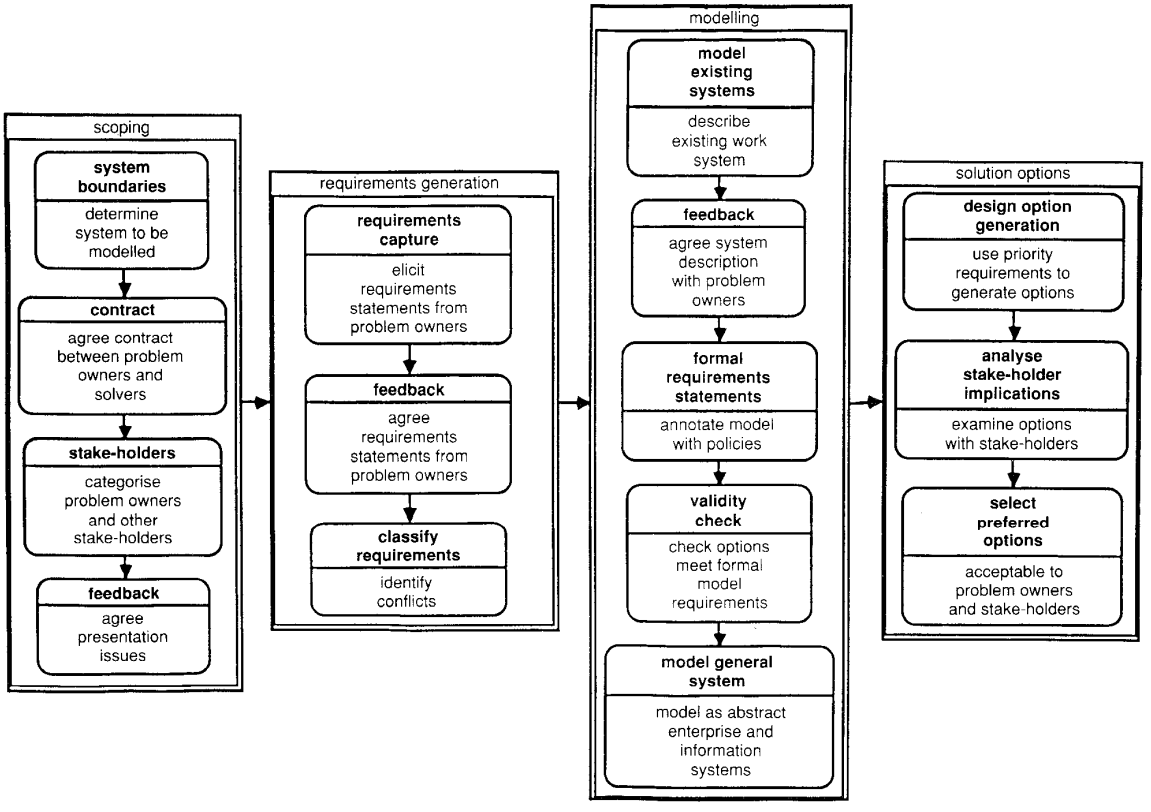

Fig. 5 ORDIT requirements engineering methodology

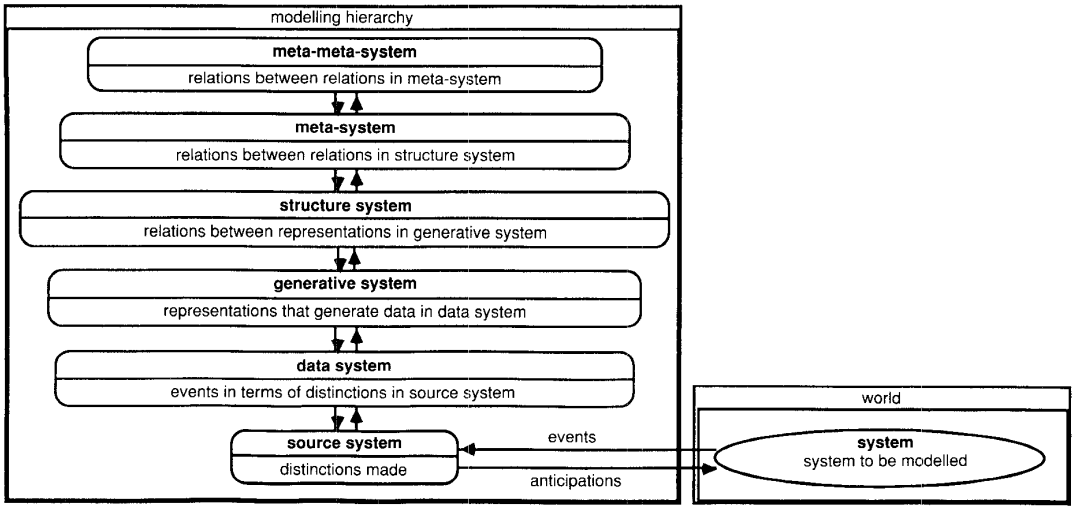

Fig. 6 Klir's modelling hierarchy

model complexity and model data approximation has proven in practice to account for a wide range of mathematical modelling techniques [27, 47]. It is interesting as an account of general adaptive processes because it makes clear the presuppositions necessary for modelling 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 on the world can be investigated [48], as can that with the socio-cultural filtering of data to improve its support of learning [49].

Klir's modeling architecture can be viewed as a general expression of the recursive processes in distinction-making at the heart of PCP and leads to a systemic model of psychological processes [49]. In addition, the analysis in these general terms is applicable to all of the different modelling areas of KADS, and it is apparent that there is useful convergence between the methodologies developed specifically for knowledge-based systems and the more general modelling methodologies of the general systems literature.

6 Modelling tools

The complexity of knowledge engineering, as illustrated in Figs. 1–6, 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 [50, 51] and may be expected to continue to be so [52].

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 illustrations can encode expertise, supplementing computational knowledge. Thus, the parallel development of hypertext and hypermedia has a substantial impact on expert system architectures and knowledge acquisition tools. Fig. 7 shows some of the features of modern document processing systems that impinge on knowledge acquisition. Documents may be acquired from many sources, displayed, reused in other documents and linked for hypertext navigation. The text in documents may also be analysed for associative clusters, and these clusters may be grouped to indicate significant concepts [53].

Fig. 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 [54]. The document analysed is on *The Technical Concept of IMS* [55] that played a major role in

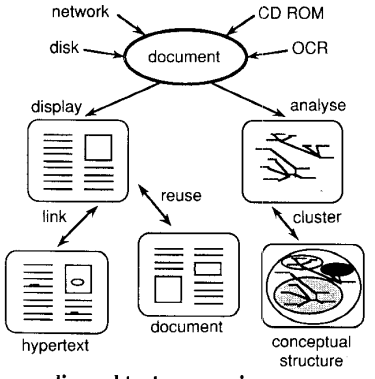

Fig. 7 Hypermedia and text processing

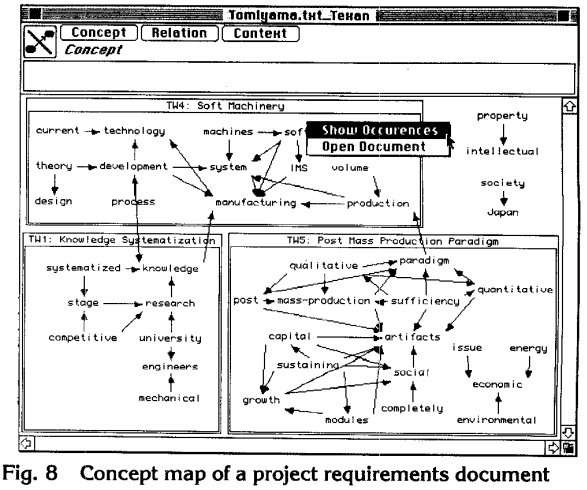

Fig. 8 Concept map of a project requirements document

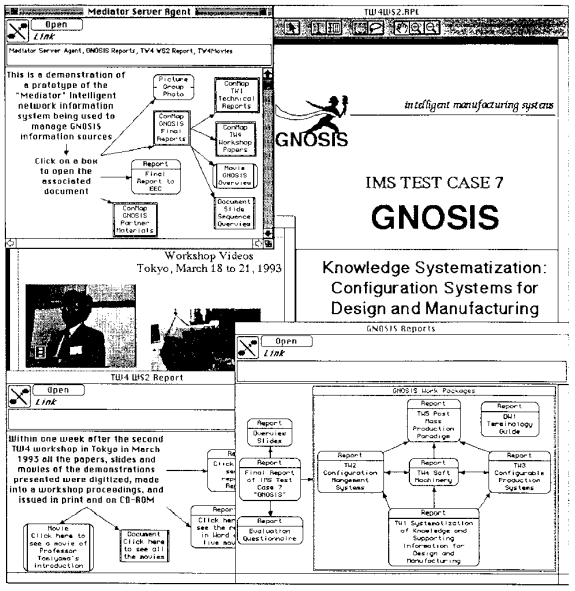

Fig. 9 Accessing the GNOSIS archives through layered concept maps

the design of the Intelligent Manufacturing Systems research programme. 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. Fig. 8 shows the links from premises to conclusions derived in this way. The tool is interactive and provides access through a pop-up 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 three loosely connected sub-components, and added the context boxes to distinguish and name these parts. The significance of these parts is that they correspond to three of the five technical work packages (TWs) of the research programme. 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 programme during its formation through amalgamation of interests with other potential proponents of IMS test cases. These packages link technically to the knowledge systematisation activities on the left of Fig. 8 but are neutral to the major issues of the IMS programme on the right. Thus, a text analysis system can structure the conceptual framework of requirements statements, and indicate areas that are inadequately represented [53].

Hypertext-based knowledge acquisition tools have been developed for use by domain experts to enter relevant case histories directly [56, 57]. They have also been used to support the knowledge engineer in structured analyses of interview material. For example, Woodward's [58] *Cognosys* supports the analysis of protocols in terms of Graesser and Clark's [59] linguistically derived 'general knowledge structures'. Other knowledge acquisition tools such as KEATS [60] 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 [61]. As 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 [62].

Fig. 9 shows an archive of some 400 MByte of multimedia data being managed through a hypermedia indexing system [63]. The system is designed for collaborative networked interaction, and the map in the upper left window is a top-level 'Server Agent' that manages a particular collection of material. The concept map in the top left of Fig. 9 is currently write-disabled, and the cursor changes to a button as the user moves the mouse over the 'Group Photo' node. Clicking at this point will display the photograph in a separate window.

The user has already clicked on the node 'GNOSIS Reports' to open the concept map shown on the lower right of Fig. 9. 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 of Fig. 9. The user has then clicked on the node 'Click here to see the report in Replica', and opened the report partially visible at the back on the right of Fig. 9. The user has also clicked on the 'Click here to see all the movies' node and opened the *KWrite* [64] 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 [65]. The use of concept maps to support systems modelling methodologies has been shown in Figs. 4–6.

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 modelling. 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 visualisation environments. A wide range of knowledge acquisition tools has 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, e.g. MOLE [66], KNACK [67], SALT [68], KEATS [60] and KRS [69].

Fig. 10 shows a knowledge structure developed in KDraw, a visual language for representation language of KL-ONE knowledge structures [70]. The problem is one of room allocation derived from an ESPRIT project [71] that was part of Project Sisyphus. The visual language is precisely defined [72]: 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 denote 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 in at the top left of Fig. 10; and a concept Æ role Æ concept triple represents a definitional role with a conceptual constraint, such as an 'organisation' has exactly '1' (cardinality constraint) 'head' who is a 'head' (conceptual constraint) near the top right of Fig. 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. Many indirect knowledge acquisition tools leave the knowledge presentation and editing to the associated performance tools as 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.

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 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 useful technique 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 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, e.g. PLANET [30], ETS [31, 73], AQUINAS [32], KITTEN [34], and KSSO [74].

Fig. 11 shows a screen from the repertory grid elicitation program KSSO RepGrid being used to elicit knowledge about factory layout planning. Two of the elicited constructs have been found to be highly matched, and the program shows the user the basis for the match and prompts 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 entered cases due to missing data.

Fig. 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.

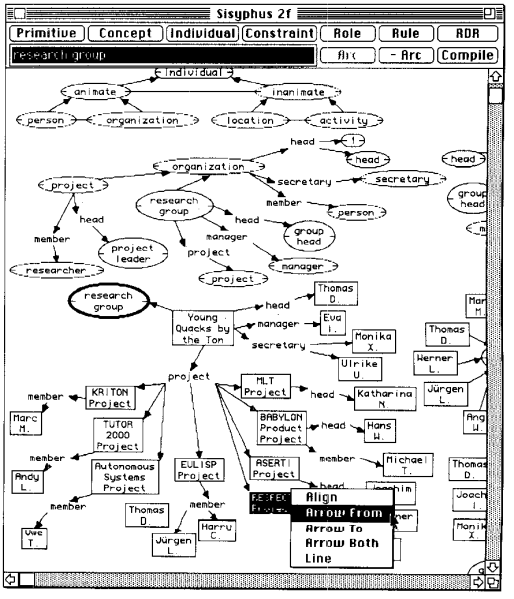

Fig. 10 Visual knowledge editing as a formal semantic net in KDraw

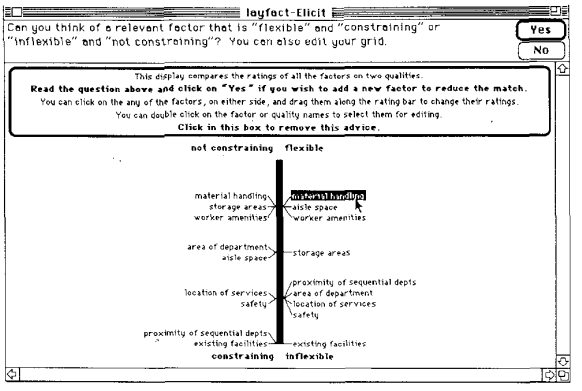

Fig. 11 Construct match being used to elicit a new element

The data from the elicited repertory grid may be modelled inductively to derive rules, and exported as a set of conceptual definitions and inference rules to an expert system shell [74]. It can also be used in a similar way to define the structure of a database with the rules providing normalisation 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 [75, 76].

7 Collaborative modelling

A common aspect of both knowledge acquisitions and requirements engineering is that the relevant knowledge is generally not available from a single person, but instead involves a process of elicitation and negotiation across a group of people. The Joint Application Development [11] and Participatory Design [12] methodologies both emphasise group processes in developing a requirements specification. The modelling tools described above were all originally targeted on individual knowledge acquisition, but in recent years many of them have been redeveloped within a computer-supported co-operative work framework to support collaborative modelling.

7.1 Group elicitation and comparison of repertory grids

The relationships between individual conceptual models have been analysed in PCP [77], and the *Socio* methodology has been developed to compare repertory grids from different people construing the same elements [78, 79]. Socio compares the conceptual models of two individuals in terms of the distinctions made and the terminology used. The two relations of similarity between distinctions and between terminology give rise to a four-way classification of constructs [75, 80], as shown in Fig. 13. *Consensus* arises if the construct systems assign the same term to the same distinction. *Conflict* arises if the systems assign the same term to different distinctions. *Correspondence* arises if the systems assign different terms to the same distinction. *Contrast* arises if the systems assign different terms to different distinctions. The usual technique for eliciting these comparisons is to have two people negotiate a common set of elements characterising a domain, each separately develop their personal constructs based on these elements, and then exchange their grids with the ratings removed and attempt to rate the elements on the other's constructs. The comparison of the exchanged grids allows consensus and conflict to be modelled, and comparison of the original grids allows correspondence and contrast to be modelled.

It is often difficult to bring together the individuals relevant to collaborative modelling in the same place at the same time, and tools have been developed for the elicitation, analysis and negotiation of conceptual models across a network [76, 81]. A recent development has been to port these tools to operate on the World Wide Web [82] in client-server mode. *WebGrid* [83] is a version of RepGrid in which the grid elicitation and analysis tools operate on a web server, and any web browser on any platform from anywhere on the Internet can participate in collaborative modelling. Fig. 14 shows WebGrid being used to develop an analysis of existing programs for implementing instructable systems with a view to determining the major dimensions of their functionality.

Grid analyses such as the FOCUS clustering of Fig. 12 are provided by RepGrid operating at the server and

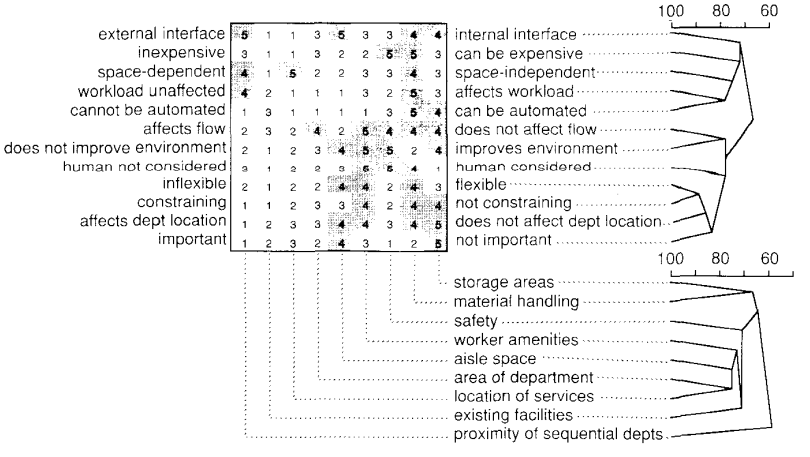

Fig. 12 FOCUS cluster analysis of a repertory grid

distinctions \ terminology	same	different
same	*consensus* individuals use terminology and distinctions in the same way	*correspondence* individuals use different terminology for the same distinctions
different	*conflict* individuals use same terminology for different distinctions	*contrast* individuals differ in terminology and distinctions

Fig. 13 Four-way comparison of constructs in terms of the distinctions made and the terminology used for them

delivered across the web as GIF files that can be displayed by the browser. Grids developed by different people may be compared to provide an analysis of the relationships between distinction and terminology as shown in Fig. 13. Fig. 15 shows such an analysis of the correspondence between constructs in the grids elicited from two people in the same development team. Such an analysis will usually lead to prolonged discussion and new insights, or the resolution of disagreement. Thus, WebGrid supports the negotiation of a combined model that expresses some of the relationships between individual models.

7.2 Group elicitation and sharing of concept maps

The drawing tool used to produce the concept maps and semantic networks in this paper has also been ported to the World Wide Web as *WebMap*, which can act as both as a client *helper* and also as a web server [84]. Fig. 16 shows WebMap being used as a local application communicating with the Netscape browser to provide the same access to hypermedia archives as illustrated in Fig. 9.

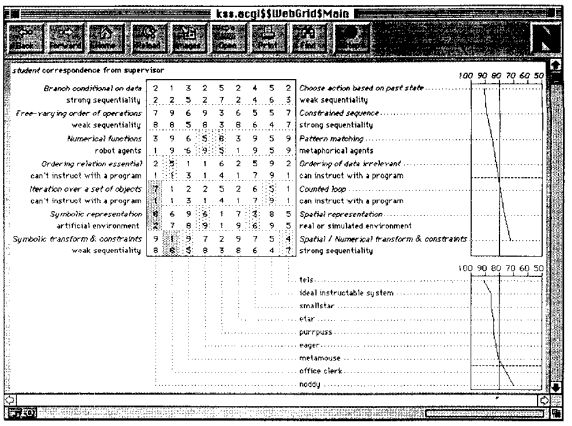

Fig. 15 WebGrid showing the correspondence between two sets of constructs

However, the archives are now distributed across multiple sites and accessed through uniform resource locators (URLs) stored in each node of the concept map, rather than by local file paths. The user has fetched the top level index to the archives as a concept map using Netscape which recognises the file type as one registered with it as belonging to WebMap and opens it in the concept mapping tool. The user has clicked on the node 'Final Report of IMS Test Case 7 GNOSIS' in the centre left of the concept map, and the concept mapping tool has requested that Netscape fetchs the associated file.

The concept mapping tool can be used locally to develop and edit maps, and then to upload them to the server again using Netscape. As shown in Fig. 17, these maps may be requested from the server as *clickable maps* in GIF format that can be directly loaded by Netscape by users who are not using WebMap as a client

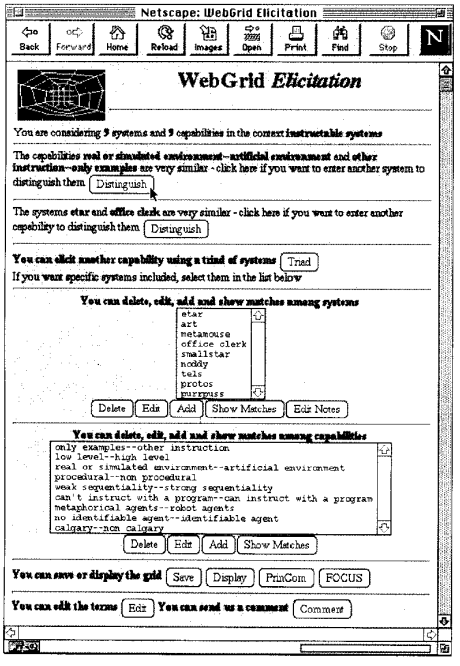

Fig. 14 WebGrid continuing display of elements, constructs and functionality

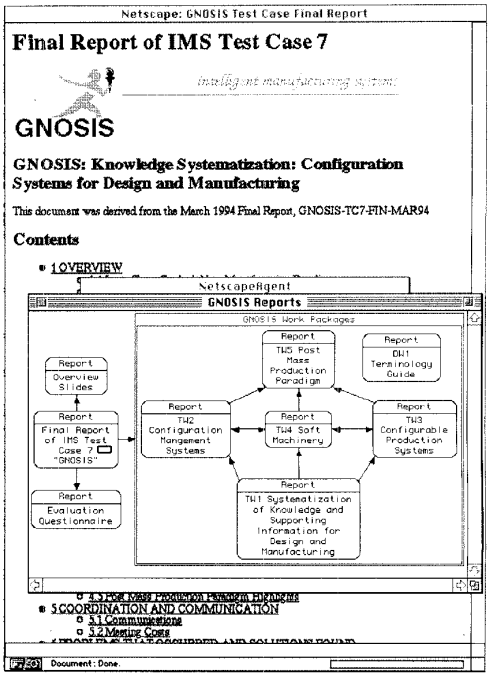

Fig. 16 WebMap used as a Netscape client helper to access hypermedia archives through World Wide Web

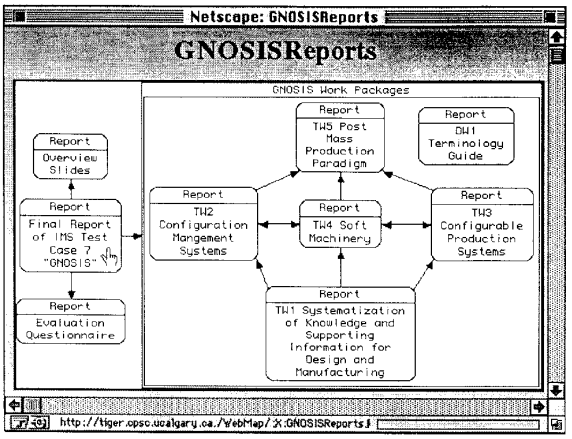

Fig. 17 WebMap as an auxiliary server delivering the concept map of Fig. 16 as a clickable image accessing the same hypermedia archives

helper. Clicking in the node in the concept map within Netscape causes the map name and co-ordinates clicked to be sent to the server where WebMap is used to decode the map, and take the same action as if the click had occurred in a local client helper. This allows archives to be made available across the Internet to Web users who do not have the concept mapping tool.

8 Related developments in requirements engineering

There are many developments in requirements engineering that relate to the knowledge acquisition techniques described in this paper. The ORDIT methodology based on soft systems methodology has been described in Section 5.3. Papers in special issues of journals and requirements engineering often address areas that overlap with those of knowledge engineering, e.g. Brun-Cottan and Wall [85] on *using video to represent the user* and Lindland *et al.* [86] on *understanding quality in conceptual modeling*. In this Section we briefly outline other relevant projects.

In 1991, the Software Engineering Institute held a workshop to investigate process-related problems in requirements engineering [87]. The overall recommendations and conclusions included:

- adopting an evolutionary, incremental approach to requirements engineering
- encouraging innovation in the acquisition process
- adopting a multi-disciplinary team approach for requirements engineering
- focusing on up-front requirements definition and validations using prototyping
- separating user interface requirements from functional requirements
- capturing major decisions and rationales

The NATURE Project [88–90] is a large ESPRIT initiative investigating novel approaches to requirements engineering. It is argued that new approaches are necessary because the environment in which requirements engineering has to operate has changed dramatically since the current methods were invented. The task of requirements engineering has moved from supporting the early phases of individual projects, to accompanying the whole life cycle of complex long-term human/machine systems in a rapidly changing organisational environment.

The NATURE framework addresses these new demands by defining a framework in which requirements engineering is a continuous process of establishing the visions of different stake-holders in a complex context. Within this framework, NATURE has developed three specific theories. The *requirements domain theory* gives advice on what context knowledge is relevant and how to organise it. The *requirements process theory* offers a unified process meta-model, in which a small set of building blocks covers a larger spectrum of process guidance strategies with more flexibility than other software process or workflow models. The *knowledge representation theory* aims at defining what domain and process knowledge to capture, and how to manage this knowledge using an effective mix of informal, semi-formal and formal representations. The practical application of this theory is illustrated in Maiden and Sutcliffe's [91] techniques for *requirements critiquing using domain abstractions*.

One of the long-standing problems addressed by the NATURE Project is that of *non-functional requirements*: how the goals of reliability, performance, accuracy, or security can be incorporated into the requirements and design process [90]. This issue is also addressed by Mylopoulos *et al.* [92], who present a process-oriented framework for non-functional requirements in terms of inter-related AND/OR goals.

Sommerville and Rodden [93] recommend the use of *ethnographic* techniques to support the integration of computer supported co-operative work with professional software engineering where there are multiple end-users co-operating explicitly or implicitly. Ethnography is one way of revealing actual, rather than formal, organisational structures, which are those that must actually be supported, and techniques have been devised for incorporating ethnographic techniques in existing methods of requirements analysis.

Loucopoulos and Champion [94] have used concept maps and conceptual graphs to capture requirements specification. They encourage the development of informal models and use these to maintain many different views about the captured concepts which enable the tracing of decisions taken by analysts and assist the identification of candidate concepts for the system.

The AMORE Project [65, 95] is concerned with ways in which large amounts of multimedia information can be visualised, stored and retrieved. Raw requirements can arise in many formats, e.g. informal technical notes, meeting notes, statements of work, interviews with customers or users, operating manuals, technical and graphical notations. AMORE is used to capture and organise such requirements in their original form, and provide the basic environment to facilitate navigation and browsing of large quantities of material.

9 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 *modelling 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 modelling and the PCP approach to human modelling processes. It is supported by general modelling methodologies such as *soft systems analysis* and the *epistemological hierarchy*, and the modelling approach is common to many software engineering methodologies.

The modelling perspective raises obvious questions as to how modelling for knowledge-based systems differs from system modelling 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. This means that 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 emphasise 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 characterised 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 modelling 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 that are designed to manage and organise the large amounts of heterogeneous data gathered in the early phases of knowledge engineering are applicable to the management and organisation 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 useful for the knowledge engineering community to widen its horizons and analyse the role of its theories, methodologies and tools in the development of any advanced information system. It would also be productive for the requirements engineering community to draw on the theories, methodologies and tools developed for knowledge engineering in the past decade, and to reuse 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.

10 Acknowledgments

Financial assistance for this work has been made available by the Natural Sciences and Engineering Research Council of Canada.

The authors are grateful to Rudi Studer, Bill Swartout and the referees for detailed comments which improved the paper.

11 References

[1] FEIGENBAUM, E.: 'Knowledge engineering: the applied side of artificial intelligence'. STAN-CS-80-812, Department of Computer Science, Stanford University, 1980

[2] HAYES-ROTH, F., WATERMAN, D.A., and LENAT, D.B. (Eds.): 'Building expert systems' (Addison-Wesley, Reading, Massachusetts, 1983)

[3] GAINES, B.R.: 'Modeling and extending expertise'. EKAW'93, Knowledge Acquisition for Knowledge-Based Systems, AUSSENAC, N., BOY, G., GAINES, B., LINSTER, M., GANASCIA, J.G., and KODRATOFF, Y. (Eds.) (Springer, Berlin, 1993), pp. 1–22

[4] QUINTON, A.: 'Knowledge and belief', *in* EDWARDS, P. (Ed.): 'The encyclopedia of philosophy: volume 4' (Macmillan, New York, 1967), pp. 345–352

[5] RUNES, D.D.: 'Dictionary of philosophy' (Peter Owen, London, 1972)

[6] SHORTLIFFE, E.H.: 'Computer-based medical consultations' MYCIN' (Elsevier, New York, 1976)

[7] DAVIS, R.: 'TEIRESIAS: applications of meta-level knowledge', *in* DAVIS, R., and LENAT, D.B. (Eds.): 'Knowledge-based systems in artificial intelligence' (McGraw-Hill, New York, 1982), pp. 229–461

[8] CLANCEY, W.J.: 'From GUIDON to NEOMYCIN and HERACLES in twenty short lessons', *in* LAMSWEERDE, A.V., and DUFOUR, P. (Eds.): 'Current issues in expert systems' (Academic Press, London, 1987, pp. 79–123)

[9] GILB, T.: 'Principles of software engineering management' (Addison-Wesley, Wokingham, UK, 1988)

[10] MAES, P., and NARDI, D. (Eds.): 'Meta-level architectures and reflection' (North-Holland, Amsterdam, 1988)

[11] AUGUST, J.H.: 'Joint application design: the group session approach to system design' (Yourdon Press, Englewood Cliffs, New Jersey, 1991)

[12] SCHULER, D., and NAMIOKA, A. (Eds.): 'Participatory design' (Erlbaum, Hillsdale, New Jersey, 1993)

[13] CLANCEY, W.J.: 'Viewing knowledge bases as qualitative models', *IEEE Expert*, 1989, **4,** (2), pp. 9–23

[14] GREENSPAN, S., MYLOPOULOS, J., and BORGIDA, A.: 'On formal requirements modeling languages: RML revisited'. Proc. 16th Int. Conf. on Software Engineering (IEEE Press, New York, 1994), pp. 135–148

[15] DOBSON, J.E., BLYTH, A.J.C., CHUDGE, J., and STRENS, M.R.: 'The ORDIT approach to organisational requirements', *in* JIROTKA, M., and GOGUEN, J. (Eds.): 'Requirements engineering: social and technical issues' (Academic Press, London, 1994), pp. 87–106

[16] MCDERMID, J.A.: 'Requirements analysis: orthodox, fundamentalism and heresy', *in* JIROTKA, M., and GOGUEN, J. (Eds.): 'Requirements engineering: social and technical issues' (Academic Press, London, 1994), pp. 17–40

[17] FORD, K.M., BRADSHAW, J.M., ADAMS-WEBBER, J.R., and AGNEW, N.M.: 'Knowledge acquisition as a constructive modeling activity', *Int. J. Intell. Syst.*, 1993, **8,** (1), pp. 9–32

[18] BOEHM, B.W.: 'A spiral model of software development and enhancement', *IEEE Computer*, 1988, **21,** (5), pp. 61–72

[19] FORD, K.M., and BRADSHAW, J.M. (Eds.): 'Knowledge acquisition as modeling' (Wiley, New York, 1993)

[20] AKKERMANS, H., HARMELEN, F.v., SHREIBER, G., and WIELINGA, B.: 'A formalisation of knowledge-level models for knowledge acquisition', *Int. J. Intelligent Syst.*, 1993, **8,** (2), pp. 169–208

[21] GAINES, B.R., SHAW, M.L.G., and WOODWARD, J.B.: 'Mod-

eling as a framework for knowledge acquisition methodologies and tools', *Int. J. Intell. Syst.*, 1993, **8,** (2), pp. 155–168
[22] COUGER, J.D.: 'Evolution of business system analysis techniques', *ACM Comput. Surv.*, 1973, **5,** (3), pp. 167–198
[23] COHEN, B., HARWOOD, W.T., and JACKSON, M.I.: 'The specification of complex systems' (Addison-Wesley, Wokingham, UK, 1986)
[24] SCHREIBER, A.T., WIELINGA, B.J., and BREUKER, J.A. (Eds.): 'KADS: a principled approach to knowledge-based system development' (Academic Press, London, 1993)
[25] GAINES, B.R., and SHAW, M.L.G.: 'Basing knowledge acquisition tools in personal construct psychology', *Knowl. Eng. Rev.*, 1993, **8,** (1), pp. 49–85
[26] CHECKLAND, P.: 'Systems thinking, systems practice' (Wiley, Chichester, UK, 1981)
[27] KLIR, G.J.: 'Architecture of systems problem solving' (Plenum Press, New York, 1985)
[28] KELLY, G.A.: 'The psychology of personal constructs' (Norton, New York, 1955)
[29] GAINES, B.R., and SHAW, M.L.G.: 'New directions in the analysis and interactive elicitation of personal construct systems', *Int. J. Man-Mach. Studies*, 1980, **13,** pp. 81–116
[30] SHAW, M.L.G., and GAINES, B.R.: 'A computer aid to knowledge engineering', Proc. British Computer Society Conf. on Expert Systems, Cambridge, 1983, pp. 263–271
[31] BOOSE, J.H.: 'Personal construct theory and the transfer of human expertise'. Proc. AAAI-84, 1984, American Association for Artificial Intelligence, California, pp. 27–33
[32] BOOSE, J.H., and BRADSHAW, J.M.: 'Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for knowledge-based systems', *Int. J. Man-Mach. Stud.*, 1987, **26,** pp. 3–28
[33] DIEDERICH, J., RUHMANN, I., and MAY, M.: 'A knowledge acquisition tool for expert systems', *Int. J. Man-Mach. Studies*, 1987, **26,** (1), pp. 29–40
[34] FORD, K.M., CAÑAS, A., JONES, J., STAHL, H., NOVAK, J., and ADAMS-WEBBER, J.: 'ICONKAT: an integrated constructivist knowledge acquisition tool', *Knowl. Acquisition*, 1990, **3,** (2), pp. 215–236
[35] GARG-JANARDAN, C., and SALVENDY, G.: 'A conceptual framework for knowledge elicitation', *Int. J. Man-Mach. Stud.*, 1987, **26,** (4), pp. 521–531
[36] SHAW, M.L.G., and GAINES, B.R.: 'KITTEN: knowledge initiation and transfer tools for experts and novices', *Int. J. Man-Mach. Stud.*, 1987, **27,** (3), pp. 251–280
[37] GAINES, B.R., and SHAW, M.L.G.: 'Hierarchies of distinctions as generators of system theories'. Proc. Society for General Systems Research Int. Conf., SMITH, A.W. (Ed.) Society for General Systems Research, Louisville, Kentucky, 1984, pp. 559–566
[38] SHAW, M.L.G., and GAINES, B.R.: 'Personal construct psychology foundations for knowledge acquisition and representation'. Proc. EKAW-93: Seventh European Workshop on Knowledge Acquisition for Knowledge-Based Systems, AUSSENAC, N., BOY, G., GAINES, B., LINSTER, M., GANASCIA, J.G., and KODRATOFF, Y. (Eds.), 1993, pp. 256–276
[39] BRADSHAW, J.M., FORD, K.M., ADAMS-WEBBER, J.R., and BOOSE, J.H.: 'Beyond the repertory grid: new approaches to constructivist knowledge acquisition tool development', *Int. J. Intell. Syst.*, 1993, **8,** (2), pp. 287–33
[40] WILSON, B.: 'Systems: concepts, methodologies, and applications' (Wiley, Chichester, UK, 1984)
[41] SHAW, M.L.G., and GAINES, B.R.: 'Eliciting the real problem', *in* WEDDE, N. (Ed.): 'Adequate modeling of systems' (Springer, Berlin, 1983), pp. 100–111
[42] MUMFORD, E.: 'Designing human systems' (Manchester Business School Publications, Manchester, UK, 1983)
[43] BLYTH, A.J.C., and CHUDGE, J.: 'The role of interaction analysis in requirements engineering'. Proc. IFIP WG 8.1 Conf. on Information System Development Process, 1993, Como, Italy
[44] DOBSON, J.E., BLYTH, A.J.C., CHUDGE, J., and STRENS, M.R.: 'The ORDIT approach to requirements identification'. Proc. 16th Ann. Int. Computer Software and Applications Conference, 1992, Chicago, Illinois
[45] STRENS, M.R., and DOBSON, J.E.: 'Responsibility modelling as a technique for organisational requirements definition', *Intell. Syst. Eng.*, 1994, **3,** (1), pp. 20–26
[46] KLIR, G.J.: 'Identification of generative structures in empirical data', *Int. J. Gen. Syst.*, 1976, **3,** pp. 89–104
[47] GAINES, B.R.: 'System identification, approximation and complexity', *Int. J. Gen. Syst.*, 1977, **2,** (3), pp. 241–258
[48] GAINES, B.R.: 'Behaviour/structure transformations under uncertainty', *Int. J. Man-Mach. Studies*, 1976, **8,** (3), pp. 337–365
[49] GAINES, B.R.: 'Social and cognitive processes in knowledge acquisition', *Knowl. Acquisition*, 1989, **1,** (1), pp. 251–280
[50] BOOSE, J.H.: 'A survey of knowledge acquisition techniques and tools', *Knowl. Acquisition*, 1989, **1,** (1), pp. 39–58
[51] BOOSE, J.H., and GAINES, B.R. (Eds.): 'Knowledge acquisition tools for expert systems' (Academic Press, London, 1988)
[52] GAINES, B.R., and SHAW, M.L.G.: 'Integrated knowledge acquisition architectures', *J. Intell. Inform. System.*, 1992, **1,** (1), pp. 9–34
[53] GAINES, B.R., and SHAW, M.L.G.: 'Using knowledge acquisition and representation tools to support scientific communities'. AAAI'94, Proc. Twelfth Nat. Conf. on Artificial Intelligence (AAAI Press/MIT Press, Menlo Park, California), 1994, pp. 707–714
[54] CALLON, M., LAW, J., and RIP, A. (Eds.): 'Mapping the dynamics of science and technology' (MacMillan, Basingstoke, UK, 1986)
[55] TOMIYAMA, T.: 'The technical concept of IMS'. RACE Discussion Paper RA-DP2, Research into Artifacts, Center for Engineering, The University of Tokyo, 1992
[56] JONES, W.P.: 'Bringing corporate knowledge into focus with CAMEO', *Knowl. Acquisition*, 1990, **2,** (3), pp. 207–239
[57] RANTANEN, J.A.: 'Hypermedia in knowledge acquisition and specification of user interface for KBS: an approach and a case study', *Knowl. Acquisition*, 1990, **2,** (3), pp. 259–278
[58] WOODWARD, B.: 'Knowledge engineering at the front-end: defining the domain', *Knowl. Acquisition*, 1990, **2,** (1), pp. 73–94
[59] GRAESSER, A.C., and CLARK, L.F.: 'Structures and procedures of implicit knowledge' (Ablex, New Jersey, 1985)
[60] MOTTA, E., EISENSTADT, M., PITMAN, K., and WEST, M.: 'Support for knowledge acquisition in the knowledge engineer's assistant (KEATS)', *Expert Syst.*, 1988, **5,** (1), pp. 6–28
[61] GOMEZ, F., and SEGAMI, C.: 'Knowledge acquisition from natural language for expert systems based on classification problem-solving methods', *Knowl. Acquisition*, 1990, **2,** (2), pp. 107–128
[62] GAINES, B.R., LINSTER, M., and SHAW, M.L.G.: 'An integrated knowledge support system'. Proc. FGCS'92, Int. Conf. on Fifth Generation Computer Systems, ICOT, Tokyo, 1992, pp. 1157–1164
[63] GAINES, B.R., and SHAW, M.L.G.: 'Concept maps indexing multimedia knowledge bases'. AAAI-94 Workshop on Indexing and Reuse in Multimedia Systems, AAAI, Menlo Park, California, 1994, pp. 36–45
[64] GAINES, B.R., and SHAW, M.L.G.: 'Open architecture multimedia documents'. Proc. ACM Multimedia 93, 1993, pp. 137–146

[65] CHRISTEL, M.G., WOOD, D.P., and STEVENS, S.M.: 'AMORE: advanced multimedia organizer for requirements elicitation'. Software Engineering Institute, Carnegie-Mellon University, Pittsburgh, 1993
[66] ESHELMAN, L., EHRET, D., MCDERMOTT, J., and TAN, M.: 'MOLE: a tenacious knowledge acquisition tool', *Int. J. Man-Mach. Studies*, 1987, **26,** (1), pp. 41–54
[67] KLINKER, G., BENTOLILA, J., GENETET, S., GRIMES, M., and MCDERMOTT, J.: 'KNACK—report-driven knowledge acquisition', *Int. J. Man-Mach. Stud.*, 1987, **26,** (1), pp. 65–79
[68] MARCUS, S.: 'Taking backtracking with a grain of SALT', *Int. J. Man-Mach. Stud.*, 1987, **26,** (4), pp. 393–398
[69] GAINES, B.R.: 'Empirical investigations of knowledge representation servers: design issues an applications experience with KRS', *ACM SIGART Bull.*, 1991, **2,** (3), pp. 45–56
[70] GAINES, B.R.: 'A situated classification solution of a resource allocation task represented in a visual language', *Int. J. Human-Computer Stud.*, 1994, **40,** (2), pp. 243–271
[71] VOSS, A., KARBACH, W., DROUVEN, U., LOREK, D., and SCHUCKEY, R.: 'Operationalization of a synthetic problem'. ESPRIT Basic Research Project P3178 REFLECT Task I.2.1 Report. GMD, Bonn, Germany, 1990
[72] GAINES, B.R.: 'An interactive visual language for term subsumption visual languages.' IJCAI'91, Proc. Twelfth Int. Joint Conf. on Artificial Intelligence, Morgan Kaufmann, San Mateo, California, 1991, pp. 817–823
[73] BOOSE, J.H.: 'Expertise transfer for expert systems' (Elsevier, Amsterdam, 1986)
[74] GAINES, B.R., and SHAW, M.L.G.: 'Eliciting knowledge and transferring it effectively to a knowledge-based systems', *IEEE Trans.*, 1993, **KDE-5,** (1), pp. 4–14
[75] SHAW, M.L.G., and GAINES, B.R.: 'A methodology for recognizing conflict, correspondence, consensus and contrast in a knowledge acquisition system', *Knowl. Acquisition*, 1989, **1,** (4), pp. 341–363
[76] SHAW, M.L.G., and GAINES, B.R.: 'Group knowledge elicitation over networks'. Research and Development in Expert Systems X, Proc. British Computer Society Expert Systems Conf., BRAMER, M.A., and MACINTOSH, A.L. (Eds.) (Cambridge University Press, Cambridge, UK, 1993), pp. 43–62
[77] SHAW, M.L.G.: 'Communities of knowledge', *in* EPTING, F., and LANDFIELD, A.W. (Eds.): 'Anticipating personal construct psychology' (University of Nebraska Press, Lincoln, Nebraska, 1985), pp. 25–35
[78] SHAW, M.L.G.: 'Conversational heuristics for eliciting shared understanding', *Int. J. Man-Mach. Stud.*, 1979, **11,** pp. 621–634
[79] SHAW, M.L.G.: 'On becoming a personal scientist: interactive computer elicitation of personal models of the world' (Academic Press, London, 1980)
[80] GAINES, B.R., and SHAW, M.L.G.: 'Comparing the conceptual systems of experts'. Proc. Eleventh Int. Joint Conf. on Artificial Intelligence, Morgan Kaufmann, San Mateo, California, 1989, pp. 633–638
[81] SHAW, M.L.G., and GAINES, B.R.: 'Supporting personal networking through computer networking'. Proc. CHI'91, Human Factors in Computing Systems (ACM Publications, New York, 1991), pp. 437–438
[82] BERNERS-LEE, T., CAILLIAU, R., LUOTONEN, A., NIELSEN, H.F., and SECRET, A.: 'The world-wide web', *Commun. ACM*, 1994, **37,** (8), pp. 76–82
[83] SHAW, M.L.G., and GAINES, B.R.: 'Comparing constructions through the web'. Proc. CSCL95, Computer Support for Collaborative Learning, SCHNASE, J.L., and CUNNIUS, E.L. (Eds.) (Lawrence Erlbaum, Mahwah, New Jersey, 1995), pp. 300–307
[84] GAINES, B.R., and SHAW, M.L.G.: 'Concept maps as hypermedia components', *Int. J. Human-Computer Stud.*, 1995, **43,** (3), pp. 323–361
[85] BRUN-COTTAN, F., and WALL, P.: 'Using video to represent the user', *Commun. ACM*, 1995, **38,** (5), pp. 61–71
[86] LINDLAND, O.I., SINDRE, G., and SOLVBERG, A.: 'Understanding quality in conceptual modeling', *IEEE Softw.*, 1994, **11,** (2), pp. 42–49
[87] Proc. Requirements Engineering and Analysis Workshop, Software Engineering Institute, Carnegie Mellon University, 1991
[88] POHL, K., ASSENOVA, P., DOEMGES, R., JOHANNESSON, P., MAIDEN, N.A.M., PLIHON, V., SCHMITT, J-R., and SPANOUDAKIS, G.: 'Applying AI techniques to requirements engineering: the NATURE prototype'. ICES Workshop on Research Issues in the Intersection Between Software Engineering and Artificial Intelligence, 1994, Sorrento, Italy
[89] JARKE, M., POHL, K., DOEMGES, R., JACOBS, S., and NISSEN, H.: 'Requirements information management: the NATURE approach', *Ingenerie Systemes d'Informations*, 1994, **2,** (6)
[90] JARKE, M., and POHL, K.: 'Requirements engineering in 2001: (virtually) managing a changing reality', *Softw. Eng. J.*, 1994, **9,** (6), pp. 257–266
[91] MAIDEN, N.A.M., and SUTCLIFFE, A.G.: 'Requirements critiquing using domain abstractions'. Proc. First Int. Conf. on Requirements Engineering (IEEE Computer Society Press, Los Alamitos, California, 1994), pp. 184–193
[92] MYLOPOULOS, J., BORGIDA, A., JARKE, M., and KOUBARAKIS, M.: 'Representing knowledge about information systems', *ACM Trans. Office Inform. Syst.*, 1990, **8,** (4), p. 325
[93] SOMMERVILLE, I., and RODDEN, T.: 'Requirements engineering for cooperative systems'. University of Lancaster, UK, 1994
[94] LOUCOPOULOS, P., and CHAMPION, R.E.M.: 'Concept acquisition and analysis for requirements specification', *Softw. Eng. J.*, 1990, **5,** (2), pp. 116–124
[95] WOOD, D.P., CHRISTEL, M.G., and STEVENS, S.M.: 'A multimedia approach to requirements capture and modeling'. Proc. First Int. Conf. on Requirements Engineering (IEEE Computer Society Press, Los Alamitos, California, 1994), pp. 53–56