*Knowledge Acquisition* (1994) **6**, 243–270

# Domain-driven knowledge modelling for knowledge acquisition

HYACINTH S. NWANA

*Department of Computer Science, University of Keele, Keele, Staffordshire, STG 5BG*

TREVOR J. M. BENCH-CAPON, RAY C. PATON AND MICHAEL J. R. SHAVE

*Department of Computer Science, University of Liverpool, Liverpool L69 3BX*

*(Received 4 May 1993 and accepted in revised form 8 March 1994)*

Knowledge modelling is undoubtedly a major problem in knowledge acquisition. Drawing from industrial case studies that have been carried out, the paper lists some key problems which still dog knowledge modelling. Next, it critically reviews current knowledge modelling techniques and tools and concludes that these real knowledge acquisition issues are not tackled by them. We consider the spelling out of these problems and the fact that they are not addressed by current tools and techniques to be a major contribution of this paper. The paper strongly argues for knowledge modelling to be domain-driven, i.e. driven by the nature of the domain being modelled. The key argument in this paper is that ignoring the nature or characterization of the domain inevitably results in *knowledge imposition* rather than *knowledge acquisition* as domains get *shoe-horned* into some (current) set of models, representations and tools. After examining the nature of domains, the paper proceeds to outline an emerging hypothesis for knowledge modelling. It concludes with a specification of a tool suite for addressing the issues identified in this paper.

## 1. Introduction

This paper describes a research project carried out at the University of Liverpool between 1989 and 1991, and which is still in progress at the University of Keele. The project was sponsored in Liverpool by Shell Research Ltd (Thornton Research Center) and Unilever Research (Port Sunlight Laboratory). The overall original aim of the project in 1990 was to attempt to provide a methodology for knowledge engineering which could replace the relatively unsystematic practices then and currently being used in knowledge engineering. In the event, the project concentrated on the very earliest stages of the process in the belief that these are crucial since without secure foundations, the resulting edifice cannot be but unstable. One of the project's results is a method for knowledge modelling (or domain characterization), even though subsequent work is further clarifying our *domain-driven knowledge modelling* paradigm with a view to constructing a suite of tools to support the method.

It may be useful to briefly report some of history which led us to articulate what we consider are the key problems with knowledge modelling.

Partly based on papers presented at the European Knowledge Acquisition Workshops, Crieff, Scotland (1991) and Germany (1992).



1042-8143/94/030243 + 28$08.00/0 

### 1.1. THE NEED FOR A METHODOLOGY FOR KNOWLEDGE ENGINEERING

Our starting point was the need for a methodology for constructing knowledge-based systems (KBSs). We took the view (which we still largely hold) that KBSs are currently in a position not dissimilar from that of conventional software systems in the late 1960s—the time of the "notorious" software crisis. At that time the potential of software applications, and the demand for these applications was well established, but the process of software development was not well understood, and there were no agreed ways of measuring the quality of software. Software development was seen as an art, not an engineering discipline, and one which could be learned only via experience. As a result:

- software developments were delivered late
- software developments ran over budget
- software systems typically failed to match customer requirements
- software systems were unreliable and unrobust
- software systems were difficult to maintain, extend and enhance
- there was shortage of trained software developers, since it is hard to train people to participate in a poorly understood process.

All of these things apply currently to KBS. An early aim of our project was to develop a method which will at least address some of these issues, and in the process serve a similar role to the software engineering methodologies which contributed to the resolution of the software crisis.

Given this picture, it is not surprising that there has been an upsurge in interest in KBS methodologies, of which most notable at the time (and still) was the KADS methodology (Wielinga, Schreiber & Breuker, 1992). KADS placed, and still does to some extent, great emphasis on the task which the KBS is intended to perform. This is entirely correct, but means that the methodology is starting from a relatively late stage. As we will later argue, it is at the stage of design, rather than specification, which is recognized as an essential prior stage in conventional life cycles. Moreover, this concentration on task will bias the knowledge of the domain, both in terms of the knowledge that is elicited and the way it is structured. Finally, viewed as a methodology at the time, KBS was quite unhelpful in that it is necessary to select an interpretation model, but there was little in the way of principled guidance to assist in model selection.

For these reasons, and for the reason of not wanting to re-invent the wheel, we decided in our project to focus on what is needed to provide the stages that must be addressed before a methodology such as KADS can be applied. This would result in aspects relating to a domain to be carried out independently of domain tasks. Such a description of a domain could be used as a firm basis for the design decisions necessitated in the later stages of development. Moreover, we believed that such a characterization required the development of a theoretical underpinning that would provide the principles to enable the characterization to be assessed to guide the knowledge engineer in elicitation and to structure the fruits of the elicitation. However, from this beginnings, we have evolved our own knowledge modelling method.

### 1.2. EXPERIMENTS (CASE STUDIES) WITH INDUSTRIAL SPONSORS

The need of a method for knowledge engineering was very general; hence, we undertook many case studies with our industrial sponsors in order to further clarify specific problems which would have to be addressed by our method. As will be

realized, they all turned out to be knowledge acquisition problems.

The case studies included the following:

- modelling the domain of artificial neural networks (Paton, Nwana, Shave, Bench-Capon & Hughes 1991)
- modelling a subset of the domain of economics (Paton, Nwana, Shave & Bench-Capon, 1991)
- modelling the 1991 Sisyphus problem of rigging a sailboat (Nwana, Paton, Shave & Bench-Capon, 1991)
- modelling part of the domain of biologically motivated computing (Paton, Nwana, Shave & Bench-Capon, 1991).

More case studies have been done since. The case studies were designed to tackle different sorts of complex domains. In all cases, our industrial partners required a characterization (model) of the domain to be understandable to non-experts. Typically, they provided us with experts and some literature (books and papers). The details of the particular domains are not important here. The goal was never to construct a KBS so as not to let the task bias the modelling process; however, this did not translate to not modelling the functionality of the domain. The product of our characterization was typically a 30–50 page document of text, diagrams and pictures easily captured within some sort of hypermedia representation. The approach we evolved, called SAAGS, is described later. Encouragingly, we found in all these studies that the experts found the characterization useful in that it helped them to gain a broader perspective on their expertise, and that the resulting characterization was a useful way of communicating the domain expertise to non-experts (including, of course, the potential KBS developers).

However, the goal of these early case studies were to identify the problems which make domain characterization non-trivial. They turned out to confirm an eclectic collation of problems which other knowledge acquisition researchers have identified. They are reported in section 2.2.

### 1.3. OVERVIEW OF THE REST OF THE PAPER

Section 2 highlights outstanding problems with knowledge modelling. Section 3 critically reviews current modelling tools and techniques for knowledge acquisition in order to establish that they do *not* address these problems, hence the necessity for such research. Drawing from sections 2 and 3, section 4 restates what and where we consider the real knowledge acquisition problem to lie: in domain characterization. But before domains are characterized/analysed, we need to take a view of what domains consist of; section 5 addresses this as well as presents an approach to characterizing domains based on the mentioned characteristics. Section 6 elaborates our view on what is involved in knowledge modelling defining the roles played by mediating and intermediate representations. It also outlines an emerging hypothesis for knowledge modelling by elaborating on the key subprocesses involved: elicitation, analysis and synthesis. The concluding section 7 argues that a tool suite based on the hypothesis of section 6 will address at least some of the key problems with knowledge acquisition outlined in section 2.

## 2. Problems with knowledge modelling

### 2.1. KNOWLEDGE MODELLING

It is certainly now more in vogue to talk of knowledge being modelled than of it being transferred or captured (Bradshaw & Boose, 1989; Breuker *et al.*, 1987; Ford

*et al.* 1993; Morik, 1989, 1991, 1993; Steels, 1990; Wielinga, Schreiber & Breuker, 1992) as it certainly "is not a substance that can be stored" (Clancey, 1989). We agree with Clancey that knowledge is something an observer ascribes to a human agent in order to describe and explain recurrent interactions the agent has with its environment or the real world (Clancey, 1989). He notes that knowledge can be represented but you can never have it in hand; the representations only exist physically in an observers statements, drawings, computer programs, silent speech, visualizations, etc.; otherwise, no observation has occurred. It is also important to note a very important point highlighted by Clancey (Clancey, 1993; Sandberg, 1991): most researchers would say this/that is the knowledge, in so doing claiming the representations are isomorphic to what is in the expert's head and that they are functionally identical. Clancey correctly notes that they are not. They are just models open to interpretation. He also points out that researchers have confused representations (models) with the phenomenon we are modelling: "the map is not the territory". This cogent argument goes to further clarify the shift from talking of knowledge being transferred or captured to it being *modelled*. Newell's (1982) classic paper identified the symbol and knowledge levels: the latter is the level in which knowledge is described while the former is inappropriate to describe knowledge.

It was perhaps thought at one time that executable representations of knowledge could themselves serve as models of knowledge that would ease the tasks of building and validating systems, and would facilitate maintenance of the systems as the knowledge changed or was refined. Moreover, such knowledge was expected to be readily transferred to other applications for which it was relevant. In practice, however, executable representations have not proved satisfactory for these purposes. Inevitably the executable representations will warp the knowledge into a form which is oriented towards the machine rather than the owner of the knowledge. Such distortions in the model are necessary to introduce features of control of execution, and efficiency of execution. Moreover, the generality of the model is often compromised by being adapted for the specific task at which the system is aimed. Significantly too, experts found themselves unable to understand the notation in which the code was written, even in the case of very high level declarative languages. For these reasons the models we will discuss in this paper are intended to be at a higher level than the executable code that will be derived from them: they are intended as a mediating stage between the knowledge sources and the code. Our belief is that such intermediate representations will enable us to realise the construction, verification, maintenance and transferability gains originally claimed for knowledge based systems.

### 2.2. SOME MAJOR PROBLEMS WITH KNOWLEDGE ACQUISITION

Drawing from the case studies we carried out, we confirm/highlight the following current problems associated with knowledge modelling.

1. It can be hard for the domain expert and knowledge engineer to harmonize their mental models (Recogzei & Plantinga, 1987). The mental model is the cognitive representations in the human knowledge source which the knowledge engineer attempts to discover and model (Shaw & Woodward, 1990).

Clearly, we want the knowledge engineer and the expert to come to share a perspective on the domain. Such harmonization cannot be achieved directly as we cannot just merge the mental models together; the engineer and the expert are constantly revising their mental models using natural language (Motta, Rajan & Eisenstadt, 1990). The case studies revealed that maintaining the constantly-changing mental models is difficult to do.

2. There often exist mismatches between elicitation techniques and the structure of the problem domain. Knowledge elicitation is the process of obtaining knowledge from an expert, to produce what may be called the "raw data". Such techniques range from structured and unstructured interviews with the expert to psychologically based methods such as laddered grid or card sorting; they are reviewed in Welbank (1990). Their efficacy varies considerably depending on the following (Burton, Shadbolt, Hedgecock & Rugg, 1987):
   - the type of knowledge the knowledge engineer is trying to obtain
   - the task in hand
   - the psychological make-up of the expert.

   It was evident that many of these techniques imposed their structure upon the elicitation process and hence the domain. This is because elicitation techniques implicitly assume characteristics about the form of knowledge to be elicited. For example, multidimensional techniques including repertory grids and card sorting mainly elicit declarative knowledge while think aloud protocols primarily aim at eliciting procedural knowledge. Hence, these techniques have limited ranges of applicability and their choice must be judicious as it changes the structure of the emerging domain.

   A primary reason for this mismatch problem lies in the fact that the literature does not make clear *when* and *where* to use these techniques; e.g. even when advice is offered by commentators concerning interviewing approaches (Grover, 1983; Welbank, 1990), this still fails to overcome difficulties. Current elicitation techniques can be classified into formal vs informal approaches, direct vs indirect methods and weak (domain independent) vs strong (domain dependent) Motta, Rajan & Eisenstadt, 1990). Such a classification should be associated with, say, Burton's factors: it must be made clear what the presuppositions of these techniques are, and when and where to use them in order to avoid this mismatch between techniques and problem domain structure (and sometimes the expert). Most crucially, as we argue later in this paper, it is the nature of the domain that should guide the elicitation process or else knowledge imposition results (see also Woodward, 1989).

3. "The analyst is after knowledge, but all he gets is words" (Recogzei & Plantinga, 1987); indeed, a deluge of words. It makes it hard for the knowledge engineer to navigate and make sense of the sheer mass of information obtained including that gleaned from books, papers and the voluminous transcripts of interviews, etc. This view is supported by a comprehensive UK survey which reported that KBS developers did not consider knowledge elicitation from the expert in itself a problem; rather, it was making sense and producing a coherent organization upon the elicited data and representing it (O'Neill & Morris, 1989).

4. Despite the numerous current tools (see Boose, 1989), a major obstacle is that little guidance is available to the domain expert or knowledge engineer to help with the following tasks (Kitto & Boose, 1989):
   - classifying the application task and identifying a problem solving method
   - given the application task characteristics, selecting knowledge acquisition tools and strategies to be applied in creating and refining the knowledge base.
5. Domain modelling tends to be driven by available tools, technique and final forms of the knowledge bases, be they frames, rules, etc. It is understandable why this so easily happens; but a further fact which emerged from investigating a variety of domains was that they required different methods for representing the domain knowledge. This confirms the importance of *allowing the nature of the domain to drive the modelling process rather than* applying preconceived methods or tools. Shaw & Woodward (1990) are right when they state: "with current systems the knowledge engineer is allowed few degrees of freedom. If the current domain does not conform to the tool, the domain is sometimes "redefined" to fit the tool". We argue that letting the tool or the final form of the knowledge base drive and constrain the modelling is a backward approach as it puts "the cart before the horse".
6. An unfortunate consequence of current unstructured approaches to knowledge acquisition is that knowledge engineers often move too quickly through some of the cognitive definition and organization work, and enter the later phases of acquisition and implementation without an adequate *specification* or *understanding* of the domain. This is one of the central claims of our work which we are addressing. This is partly due to the fact that the designs and functions of many available knowledge acquisition tools were driven by implementation rather than cognitive concerns (Shaw, 1989). Naturally, this creates problems for the neglected area of knowledge base maintenance which is a real issue for practical systems. As Ford *et al.* (1993) put it: "time invested in modelling—exploring and illuminating the domain expert's implicit assumptions (as well as our own) about the structure and dynamics of a problem domain is time saved in endless retrofitting, or failure at the implementation stage" (page 20).
7. Knowledge modelling is plainly unprincipled: there is as yet no theory of how to acquire human knowledge. Part of this is again due to the fact that models tend to be close to the symbol level. But more generally, the epistemological, cognitive and conceptual foundations of knowledge modelling leave much to be desired (Bradshaw & Woodward, 1989).

This list is not meant to be comprehensive. Our ultimate goal is to address these problems in our research.

## 3. A critical review of current modelling techniques and tools

The goal of this section is to critique current modelling techniques and tools in order to establish that these real knowledge acquisition/modelling problems, mentioned in the previous section, are not tackled, but circumvented by them. It also sets the scene for our key argument that domain characterization is crucial.

### 3.1. KBS METHODOLOGIES FOR KNOWLEDGE ACQUISITION

Methodologies are necessary to facilitate the routine development of KBSs using standard methods (taking the art out of the process), to improve their quality and to train novice knowledge engineers.

First generation KBS methodologies are mainly of two types: "stage-based" approaches and prototyping approaches. (Strictly speaking, they are really *methods* rather than *methodologies* as they do not refer to their underlying foundations; however, we use the word "methodology' in this section for consistency and as they are usually referred to as such). The stage-based approaches present life-cycle definitions. For examples, Buchanan *et al.* (1983) propose a life-cycle definition including the following stages: identification, conceptualization, formalization, implementation and testing/revision; Guida & Tasso's (1989) proposal includes plausibility study, demonstration of prototype, development of full prototype, development of target system, operation and maintenance/revision. Other examples are found in Grover (1983). These methodologies bear similarities to conventional life cycle data analysis methodologies such as SSADM (Downs, Clare & Coe, 1986). Clearly, these methodologies are very general as they attempt to address the entire complex knowledge engineering endeavour. In so doing, they fail to acknowledge knowledge acquisition as a separate stage.

A second generation of KBS methodologies later emerged which started to acknowledge the complexity of the knowledge acquisition process and sees a solution in the construction, creation or modelling of expertise. These methodologies have suggested languages for conceptual modelling. They include: ontological analysis (Alexander *et al.*, 1987), Sowa's conceptual graphs (Clancey, 1985), approaches based on generic tasks (e.g. Chandrasekaran, 1985) and conceptual modelling (e.g. the KADS methodology; Breuker *et al.*, 1987).

Ontological Analysis is a methodology designed for knowledge-level analysis (Alexander, *et al.*, 1987). Knowledge-level modelling refers to the modelling of an intelligent system's behaviour independent of whatever symbols might ultimately be used to implement those behaviours in a computer, e.g. frames, semantic nets or rules. Hence, at the knowledge level, one does not talk at the level of *how* the rule interpreter works, rather, one describes *what* it does. SUPESPOONS is the representational language which Alexander and his colleagues have proposed to carry out such an analysis. Using domain equations of denotational semantics and algebraic specification, SUPESPOONS specifies domain objects in terms of their relationships and transformations; the main objective being to identify and construct an adequate knowledge representation for any given problem. However, we do not subscribe to this latter view; while such a representation may be adequate for a few domains, the majority of domains would have to be shoe-horned into it, which is antithetical to the ideas espoused in this paper.

The KADS methodology (Wielinga, Schreiber & Breuker, 1992) is one in which the authors suggest a four-layer architecture describing the domain, the type of inferences, the tasks and strategic structures. They also describe a number of domain-independent conceptual primitives used for presenting the inference and task layers. These primitives, called interpretation models in KADS, are quite similar to Clancey's (1986) and Chandrasekaran's (1985) six generic types which depict the levels where knowledge-level analysis is carried out (e.g. heuristic

classification); these are further elaborated upon later. KAD's view of the knowledge acquisition process is one of "interpretation": verbal data (e.g. from interviews with experts or textbooks) is interpreted or mapped on to other representations and structures (Hayward, Wielinga & Breuker, 1987); classical KBS development map directly from verbal data to, say, production rules with the obvious dangers of misinterpretation and under-interpretation. Although, they can not yet claim that such a set of primitives is complete, their proposal is already robust and detailed enough to be used in a number of practical problems.

Johnson (1989) has proposed modelling knowledge via systematic grammar networks (SGNs) which she espouses as a suitable "mediating representation" which mediates between verbal data and standardized knowledge representation schemes found in AI environments. Her underlying ethos lies in the thesis that premature encoding of a knowledge in a KBS-driven representational form is often a hindrance to analysis. Johnson claims that SGNs are a well-defined system based on the idea of choice and with no standard rules of application. Its proposed advantages include conceptual modelling without having to translate concepts to a knowledge representation language, allowing for changes through further acquisition as well as acting as a source of documentation.

Sowa's use of conceptual graphs (Clancey, 1985) is an obvious choice of representation language to some researchers (e.g. Recogzei & Plantinga, 1987); they stay close to the structure of the natural language used by both the knowledge engineer and the expert and they provide a clear notation for modelling. In addition, the notation is also directly machine representable. However, all it really provides is a good diagrammatic tool which allows the modelling process to be repeated until both the knowledge engineer and the expert converge on a set of diagrams which they believe adequately represents the domain. It does not guide the modelling as such.

Other AI representations, say logic or production systems (PSA), can be directly used for modelling knowledge (PSs have already been much used in prototyping approaches) but with the obvious disadvantage of the knowledge engineer prematurely concerning him/herself with issues of symbol-level modelling as Johnson warns, without any rigorous prior analysis of the domain concerned.

*3.3.1. Discussion on methodologies*

Second generation approaches with their modelling view of the knowledge acquisition process better addresses the problems of section 2.2 than the first generation ones; in fact, the latter hardly address the problem at all as earlier mentioned. Nevertheless, these second generation methodologies still suffer from many shortcomings. The guidelines they provide for conceptual modelling are still fuzzy. These approaches still do not fully address the issue of harmonizing the expert's and the knowledge engineer's mental models (Recogzei & Plantinga, 1987); neither do they suggest where and when to use the various elicitation techniques nor what tools to use. KADS is an exception. KADS has advantages over the others: it is principled, it has reusable models and a good toolkit, and it is emerging as the *de facto* standard for knowledge acquisition, at least in Europe.

However, these methodologies and modelling approaches can only be said to *prescribe* some domain conceptualization phase, though they themselves mainly

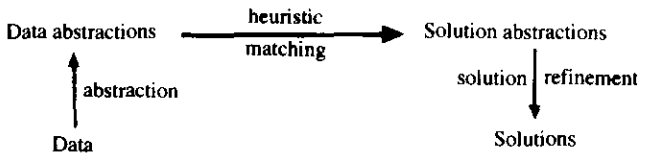


FIGURE 1. Heuristic classification problem-solving method.

provide means/languages, and some domain ontologies (in the case of KADS) for knowledge modelling.

### 3.2. A SELECTIVE CRITICAL REVIEW OF KNOWLEDGE ACQUISITION/MODELLING TOOLS

When developing a KBS, a requirements analysis phase is necessary to identify the task that the KBS will perform. Traditionally, knowledge engineers construct a model of the system's proposed behaviour which corresponds to the engineers' theory of how the expert solves problems. This modelling activity is what Newell (1982) refers to as *knowledge-level* modelling. Newell's work has heavily influenced most of the knowledge acquisition tools developed to date. Other investigators have built on this work to characterize generic tasks/methods of problem-solving that are independent of a specific inference engine and of specific application areas (Clancey, 1986; Chandrasekaran, 1985). Hence, a knowledge-level description of a system identifies its abstract data and inference types and its generic control structure. Clancey (1986) identifies heuristic classification and heuristic construction as two fundamental examples of such generic problem-solving methods. The former is a common method in which concepts are heuristically related using a process of data abstraction, heuristic matching and solution refinements [see Figure 1 (adapted from Clancey, 1986)]; this method is suitable for problems in diagnosis, repair, catalogue selection or skeletal planning.

The latter (i.e. heuristic construction method) constructs solutions by generating complete solutions or assembling them from components while satisfying constraints; this is most suited to synthesis application tasks (design, configuration, scheduling, etc.). Chandrasekaran (1985) also describes six generic tasks in knowledge-based reasoning which, to some degree, resemble the problem-solving methods proposed by Clancey. They include classification, state abstraction, knowledge-directed retrieval, object synthesis by plan selection and refinement, hypothesis matching, and assembly of compound hypotheses for abduction. Like Clancey, Chandrasekaran views these generic tasks as problem-solving methods which can be combined to perform knowledge-based reasoning for an application task. The influence of such work is evident in the overview of the tools that follow.

Numerous tools currently exist which were designed to support the knowledge acquisition process, even though most of them are just prototypes. They could be classified as (thanks to Marc Linster):

- task-specific tools
- problem-solving-method-specific tools
- reportory-grid-based elicitation tools
- machine-learning tools
- general tools.

*3.2.1. Task specific tools (model-extension tools)*

These are also referred to as role-limiting tools. OPAL (Musen *et al.*, 1987) is an example in this category. It is an interactive program that acquires new cancer treatment plans for an expert system called ONCOCIN (Shortliffe *et al.*, 1981)—a program that provides therapy advice to physicians who take care of cancer patients. Expert oncologists use OPAL to descrive new chemotherapy regimens for ONCOCIN by filling out graphical forms and by drawing flowchart diagrams on a workstation display. PROTÉGÉ (Musen, 1989) is the interactive graphics program that assists knowledge engineers to create general models of application tasks that can be solved with the problem-solving method of skeletal-plan refinement (Friedland & Iwaski, 1985), which the expert fills and refines. OPAL uses *task-based* conceptual models. (Note: this is not to be confused with the concept of "generic tasks" mentioned earlier.) OPAL's tasks model includes domain-specific concepts, e.g. chemotherapy, drug, toxic reaction, lab test, etc.) Expert physicians with little computing experience have successfully used these models to enter specific cancer treatment plans. The generic task (or problem-solving method) is skeletal-plan refinement.

Knowledge acquisition can be viewed as consisting of two related phases: constructing a generic task model—i.e. creating an *intention* of the proposed system's behaviour; and filling in the domain-specific content knowledge that is consistent with the general task model—i.e. creating *extensions* (Musen, 1989). PROTÉGÉ falls in the former category (i.e. model building tool) while OPAL falls into the latter (i.e. model-extending tool).

*3.2.2. Problem-solving-method-specific tools*

TEIRESIAS (Davis, 1979; Davis & Lenat, 1982) is a classic example in this category. It is a system devised to help with the development and maintenance of a large knowledge base for a particular expert system shell—the EMYCIN shell (Van Melle, 1979). It is basically a failure-driven tool in that interactive transfer appears in the context of discovered shortomings in a knowledge base. For example, when the expert system fails to function as expected, TEIRESIAS applies its "rule models" to detect missing information, helps place new knowledge into existing rule structures, and assists in changing these rule structures. The problem-solving method it supports is a form of heuristic classification (Clancey, 1986). TEIRESIAS operates (elicits knowledge) at Newell's (1982) symbol level; users must appreciate both the nature of the representations used to encode knowledge in the target expert system and the consequence of applying particular inference mechanisms to these symbols. Users here must understand how expert systems work. TEIRESIAS was never actually used by expert physicians for whom it was intended for this reason and motivated the development of OPAL. ROGET (Bennett, 1985) is another classic tool which conducts a dialogue with its user asking about the *problems* to be diagnosed, about the *causes* of those problems and about *data* to confirm/refute those causes and problems. The resulting knowledge-level specification was then translated and used to develop EMYCIN-based expert systems. ROGET's problem-solving method was thus some form of heuristic classification (see Figure 1).

Another example in this category is SALT (Marcus, 1988). It is based on the propose-and-revise method of heuristic construction: it constructs solutions to a

problem by successive revisions. Other examples include MOLE (Eshelman, Ehret, McDermott & Tan, 1987) and its predecessor MORE (Kahn, Knowland & McDermott, 1985) which are both based on an instantiation of the heuristic classification method called cover-and-differentiate. The last two methods are called *role-limiting methods* as they strongly constrain knowledge that needs to be gathered to do a particular task. TEIRESIAS is different from all the others mentioned in this section in that it is the only one whose problem-solving method is quite implicit in its code; the rest have explicit conceptual models of problem-solving.

*3.2.3. Reportory-grid-based tools*

ETS (Expertise Transfer System) is an interactive grid-based technique for eliciting knowledge (Boose, 1985). It automates psychiatric interviewing techniques/theories that were originally devised by therapist George Kelly (1955) to learn how people make distinctions of the world; Kelly's theory is called the Personal Construct Theory. (A construct is defined as a bipolar dimension which brings out the similarity of a set of elements and the difference of this set of elements from other elements). ETS uses a structured dialogue with an application expert to solicit the elements of the solution set (the possible classifications that may apply to a given case) and the features that may be relevant in arriving at a classification. Thus, it elicits conclusions with their similarities/differences in order to establish traits. By rating pairs of traits, it constructs a rating grid from which it infers an entailment graph which reveals the semantic distance between concepts and possible implicational relations. IF-THEN production rules are then generated (entailed and induced) from this graph. The biggest problem with ETS is that it elicits only simple classifications from the experts; it has difficulty expressing causal, procedural and strategic knowledge. The classification associations produced are also frequently spurious.

AQUINAS (Boose & Bradshaw, 1987) is the successor to ETS which improves on some of the latter's limitations, mainly in representation and reasoning. It allows elicitation from multiple experts and stores it in a hierarchical network of repertory grids. This knowledge can be examined and refined using tools that do clustering, similarity analysis, implicational analysis and consultation testing. These tools use techniques to analyse information in grids and suggest ways to refine the knowledge bases. Both ETS and AQUINAS generate operational knowledge bases for a number of expert system shells (e.g. KEE, OPS5, etc.) from the common internal representation (Boose, 1985).

Other tools based on the repertory grid interviewing techniques include PLANET, KITTEN (Shaw & Gaines, 1987) and NEXTRA (Rappaport & Gaines, 1988). NEXTRA is a knowledge acquisition toolbox based on an extendible set of techniques developed for the knowledge support system KSS0 (Shaw & Gaines, 1988). The NEXTRA technology encompasses elicitation tools, visual analysis and display tools, group comparison tools, inductive tools, and knowledge base generative tools. NEXTRA creates a knowledge base for use by the performance system NEXPERT. The KSS0 system, which subsumes the NEXTRA knowledge acquisition toolbox, supports a multi-expert grid-based elicitation recognizing consensus, correspondence, conflicts and contrasts (Shaw & Gaines, 1988).

KRITON (Diederich, Ruhmann & May, 1987) combines repertory grid interviewing and protocol analysis to build knowledge bases. We critique such tools later.

*3.2.4. General tools*

This refers to toolkits that attempt to provide a comprehensive range of tools to help the knowledge engineer bridge the gap between the expert's knowledge and some final runnable system. KEATS, developed by British Telecom and the Open University (Motta *et al.*, 1988; 1990) excellently falls into this category. It aims to provide complete life-cycle support for knowledge engineers beginning with knowledge elicitation, through problem conceptualization, knowledge encoding and debugging. It provides useful enhancements to other modern shells, toolkits and enviroments for knowledge engineering (e.g. KEE, ART, etc.) including a semi-automated transcript analysis facilities and a sketchpad on which the knowledge engineer may draw a free-hand representation of a domain. It also has a hybrid representation formalism that includes a frame-based language and rule interpreter.

The KEATS system comprises several tools including ACQUIST (a hypertext-based domain conceptualization tool). KEAT's theory is mentioned only in passing: it may be typical of AI systems where the theory and the system are one and the same (Anjewierden, 1987). Another example is SHELLEY (Anjewierden, Wielemaker & Toussaint, 1992) which is the toolkit that supports the KADS methodology.

*3.2.5. Machine learning tools*

Learning is without doubt one means of knowledge acquisition (Chandrasekaran, 1989). Machine learning can facilitate knowledge acquisition by transforming knowledge that is in a readily available form, such as examples, into a more useful form, such as diagnostic rules. If experts could simply show a machine what to do, rather than program it, then the knowledge acquisition problem would be solved (Chandrasekaran, 1989)—at least, this is the theory.

ID3 (Quinlan, 1983) creates decision trees from preclassified training examples using a divide-and-conquer approach. AQ11 (Michalski, 1983) induces rules from sets of positive and negative training examples. LEAP (Michel, Mahadevan & Steinberg, 1985) uses apprenticeship learning to learn steps in VLSI design by watching experts solve problems; it is, however, not a stand-alone system. On some level, other tools already mentioned could be viewed as learning tools (e.g. ETS/AQUINAS could be seen as learning via induction; admittedly, there is a difference in the way the latter functions compared to say AQ11). A more recent development of a machine-based knowledge acquisition system is MOBAL (Morik, 1993).

However, "for any application of machine learning to knowledge acquisition, the knowledge engineer has to set up the learning problem for the induction algorithm, design a representation for examples and generalizations, define all the terms in the language(s), encode a set of training examples in the representation and provide background knowledge (Michalski, 1983) that guides the induction algorithm to choose the right generalizations from the potentially infinite set of possibilities. This can require significant knowledge engineering effort, especially if the task is more complicated than simple classification" (Gruber, 1988, page 583).

*3.2.6. Discussion on tools*

A range of tools for knowledge acquisition has just been explored. As may have been recognized, quite a number of them are similar in terms of their functionality. It may be necessary to start off with a word of warning about tools in general. Most tools are very seductive, but users of them must realize that they carry along with them assumptions which are not normally made explicit. Users must be aware of *what* they are using and *when* to use it in addition to the usual *how* to use these tools reported in papers describing them. For example, ETS/AQUINAS really apply to declarative domains and are not of much good to expressing causal, procedural and strategic knowledge. Hence some preanalysis needs to have been done to suggest the use of ETS/AQUINAS. OPAL requires the domain to have been analysed to realize that a skeletal-plan refinement problem-solving method would be applicable. Indeed, most of these tools implicitly implement knowledge-level problem solving methods.

On role-limiting method-based tools, Gruber writes: "the key to the success of many of these tools is that they use knowledge about the task (as in OPAL) and/or a particular problem-solving method (as in ROGET, MOLE, SALT, etc.) supported by the architecture of the expert system (McDermott, 1986). This has two primary benefits for knowledge acquisition: first, the interface between the human and machine can be structured to acquire knowledge relevant to the task and only elicit knowledge in the form useful to the method (e.g. OPAL); second, since it is designed to work with a specific problem-solving method, and therefore knows how the method applies domain knowledge, the tool can analyse the user's input for completeness and consistency. In effect, the knowledge acquisition tool reduces the task of building a knowledge base to the task of instantiating pre-designed and well understood representations for a specific domain" (Gruber, 1988, page 582).

Thus, these tools all require knowledge-level analysis to be performed in advance of their implementation. Furthermore, users must know *a priori* that the particular method of problem-solving built into the tool can be applied to the task they wish the target KBS to address.

*3.2.7. What then is our point?*

The point of all this is that without judicious use, these tools may compound the knowledge acquisition problem because the wrong tool, problem-solving method, task choice or elicitation technique could be used on the wrong domain with disastrous results. Tools are powerful in what they are good at and designed for and their success, if rightly used, can not be overestated. However, they still do not tackle the problems identified in section 2.2 even though admittedly, they could contribute to their solution. For example, they do not address the problem of harmonizing the expert's and the knowledge engineer's mental models. Also, there is little guidance available to the domain expert or knowledge engineer to help with: classifying the application task and identifying a problem-solving method or, given the application task characteristics, selecting knowledge acquisition tools and strategies to be applied in creating and refining the knowledge base. Kitto & Boose (1989) have gone some way in addressing the latter problem with their ongoing work on the AQUINAS Dialog Manager. Admittedly also, toolkits like SHELLEY and KEATS provide the knowledge engineer with tools to make sense of a domain.

## 4. Restatement of the knowledge acquisition problem: domain characterization

We contend that many of the problems associated with knowledge acquisition will not be answered by developing more knowledge modelling languages/methodologies or by developing more tools; even the current trend of grouping some of these tools into toolkits and workbenches in an unprincipled manner, is unlikely to do the trick. It will only be overcome by deliberately attempting to tackle the root causes such as those identified in section 2.2. We have been working on a project called MEKAS, an abbreviation of methodology for knowledge analysis. As noted earlier, the need for this kind of research has emerged from a variety of industry-sponsored knowledge acquisition projects based at the Universities of Liverpool and Keele and is further supported by the requirements of two major industrial collaborators. Drawing from earlier sections (section 2.2 mainly), in our MEKAS projects, we sought (and still seek) to address the following problems:

1. developing techniques to enable the expert(s) and the knowledge engineer(s) to harmonize their mental models as suggested by Recogzei & Plantinga (1987)
2. providing techniques of analysing a problem domain so as to reveal *what* elicitation techniques/tools to use *when* so as to address the representation mismatch problem of section 2.2
3. develop techniques to navigate and make sense of the sheer mass of information involved in knowledge acquisition. For this, we draw some knowledge from the literature of what has been achieved thus far with methodologies such as KADS and Ontological Analysis
4. develop techniques which ensure that the nature of the *domain* that should guide the knowledge modelling process: the final form of the knowledge base should emerge from a characterization of the domain
5. develop a method which ensures that knowledge engineers do *not* move too quickly through some of the cognitive definition and organization work in order to enter the latter phases of acquisition and implementation without an adequate *specification* or *understanding* of the domain
6. develop a more principled method for domain modelling.

The approach to knowledge analysis and modelling we are developing attempts to address these issues which we consider all contribute to the knowledge acquisition problem. We contend that part of the solution to these problems is revealed as we make distinctions between *knowledge analysis* (as we see it) and *knowledge-level analysis*. A key achievement of our work would be to highlight this distinction while, at the same time revealing their complementary natures. It is important here to clearly disamiguate these two. The approach to knowledge analysis which we are evolving addresses the six issues raised above. Knowledge-level analysis emphasizes three aspects: the function of the model, the organization of the knowledge, and the control strategy (Schreiber *et al.*, 1988). Thse constitute the problem-solving method or generic task. Analysis in this case reduces to mapping a task model to a problem-solving method. This is normally called the *design* problem (Schreiber *et al.*, 1988). The important issue of note here is that knowledge-level analysis really concerns design of the KBS which necessitates that some domain

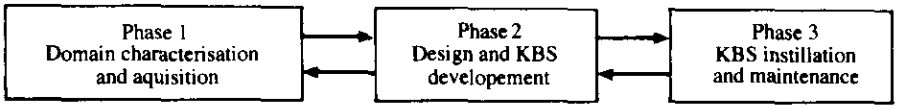


FIGURE 2. Phases of KBS development.

analysis/characterization should have earlier been done to reveal the tasks of the domain. Woodward (1990) acknowledges this fact when he notes that "they [i.e. problem-solving methods/generic tasks] are extremely rigorous and useful frameworks but are not required at the domain organization level of analysis".

Hence, knowledge analysis in our view, really concerns the characterization of a domain. As shown in Figure 2, it is really the phase 1 activity in the development of KBSs. Our contention is that the root causes of knowledge acquisition problems reside in the inadequate attention (some may go as far as to say, neglect) given to this phase by researchers as testified by the literature. Many authors suggest a feasibility analysis where issues of domain boundary, experts availability, stability of procedures in domain, recognizing the commitment in time and funding and consultation of users of the system (e.g. Parsaye & Chigwell, 1988; Martin & Oxmann, 1988) should be investigated. However, these are only general guidelines and do not address domain characterization; specific techniques which address critical issues like knowledge content, organization and structure are needed to make critical decisions before concentrating on the expensive effort of the second phase of knowledge engineering.

Domain characterization is usually done via interviews between experts and the knowledge engineer and involves organizing the knowledge (raw data) gained from human experts, the relevant literature, manuals, journals and other sources (e.g. examples, case histories) into some domain organization. It is the stage which Woodward (1990) refers to as domain definition and we concur with him when he states that it is a crucial stage in knowledge engineering which "structures the domain to isolate and organize key areas of content, then presents the knowledge in a manner that feeds more specific knowledge acquisition tools". In the approach to characterization which we are developing, we are seeking to provide, as comprehensive as possible, a description of the static and dynamic nature of a domain. Furthermore, it also makes better software-engineering sense to produce a detailed characterization at an early stage so that problems that may arise later on, concerned with maintenance or extensibility, are minimized (see e.g. Bench-Capon *et al.*, 1993).

### 4.1. CURRENT TOOLS/METHOLOLOGIES AND THE NECESSITY FOR DOMAIN CHARACTERIZATION

We further concur with Woodward (1990) when he notes that all knowledge acquisition procedures assumes domain characterization in a manner which is conducive to starting KBS activity. It also assumes that decisions on boundaries of the domain have been identified; indeed, it is normally assumed in these papers that the domain organization has been completed (Waterman, 1986).

Furthermore, powerful state-of-the-art tools like AQUINAS (Boose & Bradshaw, 1987) and KSS0 (Gaines, 1987) require primitive elements and constructs. They also

assume a certain level of specificity and granularity which presumes *a priori* domain characterization or content analysis. Using any particular elicitation technique/tool, as has been earlier noted, would normally require some *a priori* domain analysis or characterization.

It has also been established in earlier sections that other tools, e.g. SALT, MOLE, ROGET, also require the users to know *a priori* that the particular problem-solving method built into the tool can be applied to the task they wish the target KBS to address. For example, SALT requires the expert to slot in a variety of types of information in the form of events, objects, names and formulae to suit its propose-and-revise problem-solving method. Also, NEXTRA requires specifications of entities, attributes and constructs within a domain. Clearly, these tools drive the expert's (user's) thoughts in a certain fashion that limits the scope of the domain. Gaines (1994) correctly notes that the greater the assumptions about the knowledge implicit in the tools from the knowledge the user has, the worse will be the model of the domain knowledge elicited. Section 2.2.1 also noted that current methodologies remain fuzzy when it comes to conceptual modelling. For example in KADS, there are no clear guidelines as to the choice of some interpretation model. Indeed, KAD's main strengths are in its provision of possibly the most comprehensive set of interpretation models. These models really constitute design primitives; hence, they address more of phase 2 process of Figure 2 (i.e. design of the artifact) than of phase 1 since some *a priori* analysis needs to be done to make use of them. This same argument also applies to other generic task based approaches (e.g. Chandrasekaran, 1985; Clancey, 1986). This importance of domain characterization cannot, therefore, be overstated.

In effect, the motivation of this paper is to provide or work towards a more principled approach to knowledge modelling which is totally domain-driven. The key argument in this paper is that ignoring the nature or characterization of the domain inevitably results in *knowledge imposition* rather than *knowledge acquisition* as domains are *shoe-horned* into some set of preconceived models, representations and tools. This paper draws heavily from our work in MEKAS (Bench-Capon, Coenen, Nwana, Paton & Shave, 1993) but also unashamedly from the work of many other researchers in knowledge acquisition. Since it is much based on other researchers' work, we seek to provide some clarification and extension to the theoretical framwork. We strive to achieve a principled (not formal) basis to our mediating/intermediate representations by postulating a hypothesis of knowledge modelling. In so doing, we hope to make the process less a matter of inspiration than of technique.

Having argued that many of the problems with knowledge modelling lie in domain characterization, the next section outlines our view as to the nature of domains.

## 5. Towards a theory of domains

We believe that knowledge in domains is extremely complex. A basic assumption of our theory is that humans construct models in order to deal with the complexities of real world domains. The philosophical and cognitive justification for this is discussed more fully in Paton, Nwana, Shave, Bench-Capon & Hughes, 1991 and Paton, Nwana, Shave & Bench-Capon, 1991. We have identified seven fundamental or

top-level characteristics of a domain (Bench-Capon *et al.*, 1993). We briefly summarize them below.

*Theory:* the conceptual framework used to construct and maintain a domain. An appreciation that domains have theories can help a knowledge engineer gain valuable insights on a domain. This can help the engineer by providing a philosophical foundation which is used to anchor the techniques and tools used in representation and modelling. Of course, theories are fallible, approximate and socially constructed.

*Metaphor:* the language used to talk about one thing in terms of something else. The role metaphors play in domains should not be underestimated; metaphors are pervasive. They certainly give the engineer insights to the ontology, structure and functionality of domains.

*Metatheoretical constraints:* this includes key concepts such as time, space and causality which provide ontological details about the nature of domains.

*Relations with other domains:* this characteristic overlaps with metaphors in that it provides a vocabulary for talking about one thing in terms of another. It can also provide details related to the history of the domains as well as underlying models of how the domain is constructed and/or the types of problem it can deal with.

*History:* the evolution of the domain in time. Domains are not static, they change in time. An appreciation of the history of the domain, and in particular, the expert's own perspective, can be very helpful in clarifying other top level characteristics and in elaborating the theory. As Gaines (1994) suggests, the fact that domains have theories and history also says much about the nature of theories. It is also valuable to be able to guess how the domain could develop because this may influence decisions about the viability of a KBS.

*Structure:* the parts, relations and organization of the domain in relation to models.

*Purpose:* the problems which the domain addresses in terms of their solutions.

We argue that in domain-driven knowledge acquisition/modelling, at least some of these characteristics should be exploited. We have exploited them in our SAAGS approach (see below).

### 5.1. THE SAAGS APPROACH

SAAGS (Specification-Anticipation-Acquisition-Generation-Specification) is a four-stage, iterative, cyclical process which receives as input a loose specification and outputs a domain characterization (which is naturally only a model). Since history is an inseparable part, the domain characterization is also prone to change all the time. The outputs ("knowledge structures') of the approach include a domain vocabulary, domain glossary, domain structure, purpe of domain (in terms of goals and tasks), idea that relate to causality and time, history of the domain as well as metaphorical and theoretical aspects of the domain. These correspond to the characteristics of domains we have identified above.

In SAAGS (as in all knowledge modelling), analysis and elicitation are very much intertwined. We contend that if details of domains including structure, theories, metaphors and purpose are anticipated, then later acquisition (elicitation) sessions can be better structured and also reveal more of the real nature of the domain; hence the reason for *anticipation* before *acquisition* in SAAGS. The *generation* (of

models) phase follows. Note that the generation phase may generate computer-based models (e.g. a limited scope system). This way, the generation phase also serves as a testing phase of each cycle. The cycle returns to *specification* but now a great deal has been learnt and to some degree structured. The goal of SAAGS is to produce a domain specification that is coherent, comprehensive, consistent and relevant and the cycle continues, though sometimes by-passing certain stages, until an acceptable characterization is reached. Knowledge acquisition is only one of four stages and is surrounded by stages concerned with the processing of frameworks of knowledge. We believe these other three stages are essential if we are to avoid the knowledge acquisition bottleneck and conceptual superficiality.

All knowledge engineers go into an elicitation session with some expectations of the domain. By providing a systematic basis to such "anticipations", we hope to make knowledge analysis, and hence knowledge acquisition, less a matter of individual inspiration. We emphasize that anticipitation does not present the knowledge engineer with sufficient information to characterize the domain, but rather helps him/her structure the acquisition process and provides certain expectations on the knowledge to be obtained and a framework into which elicited knowledge can be incorporated. Hence, we let the analysis drive the elicitation process, suggest which knowledge acquisition/elicitation tools/techniques to use as well as suggest various intermediate representations to model the knowledge.

### 5.2. APPROACH IN PRACTICE

SAAGS is at present non-automated. In practice, it is used as follows. A client presents a knowledge engineer (KE) with an input specification (e.g. a request to explore the feasiblity of a KBS relating to some domain) and a decision is made whether it is sufficient. The first step is to precisely establish a working vocabulary for the domain of discourse. If the KE knows little about the domain, an initial simple *acquisition* phase is done, possibly involving no more than reading though the contents, and abstracts of relevant papers/books listing the major words used. The KE then *generates* a preliminary list of concepts, metaphors, and hypotheses some relationships both within the domain and with other domains. The *specification* is refined, identifying the main top level concepts in the domain, and the KE then *anticipates* likely metaphors, purpose, structure, theories and other domains to which it is related. This stage hence involves more than the "homework" done in other approaches to knowledge acquisition.

The KE now organizes the first session with the expert with the goal of establishing some kind of rapport with him/her, casting a wide net over the domain in order to appreciate its boundaries (as the expert sees them) through eliciting the domain's history, purpose, metaphors, theories, etc. At this stage the questions posed to the expert are thus quite *grand tour.* It is important to note here how anticipations have already guided the KE to ask the sort of questions which elicit aspects of domains proposed including metaphor, theory, relations to other domains, history and so forth. This session thus uses both structured and unstructured techniques. The audio-tape record is transcribed and analysed.

Primary (paper) models are *generated* from the analysis including a vocabulary of discourse, list of global metaphors, concept map of related domains, list of sortal types and properties/attributes and preliminary relations between the latter. At this

stage, the anticipated details are compared with the generated models. The differences between them furnish further questions for the next session with the expert. The *specification* is again refined; emerging outputs should include: organization of the different sortal types and their properties, relations between the latter and global metaphors, identification of key tasks (relating to the domain's functionality), relations between objects, relations with other domains. The emerging theories and metaphors should structure the generated models (paper and/or computer-based) as well as the kinds of representation of the specification/characterization and hence the domain. The cycle iterates in this manner. After each iteration the cycle returns to *specification* but a great deal has been learnt and to some degree structured. The sessions with the expert become more structured as wrong anticipations are cleared up and details are sought. Hower, some issues which were structured in earlier sessions maybe found to be inappropriate at later sessions, thereby necessitating backtracking to earlier cycles. In the worst scenario, the whole process may have to be begun from scratch. The process ends when the KE and expert mutually agree on the goal specification.

## 6. A hypothesis for knowledge modelling

What then is involved in Knowledge Modelling? It is important to address this question because as Morik (1991) points out, the terms "knowledge" and "modelling" are used with different meanings in the literature corresponding to different views of knowledge acquisition. Figure 3 serves to illustrate what knowledge modelling entails. Drawing from our experiences, knowledge modelling or analysis, really concerns the characterization and acquisition of the knowledge in some domain. It is the first phase activity in the construction on knowledge based systems. As Figure 3 shows, it involves the iteration of the key sub-activities of elicitation, analysis and synthesis culminating in some intermediate representation models for the domain. The elicitation–analysis–feedback loop serves to emphasize the new cooperative or constructive view of knowledge modelling (Morik, 1989, 1991, 1993). Also note that this loop is central to the SAAGS approach of the last section.

It should only be after such a modelling phase that later phases of KBS development and implementation should proceed.

### 6.1. MEDIATING AND INTERMEDIATE REPRESENTATIONS

In knowledge modelling, the resulting representations of knowledge fall under two categories: mediating and intermediate representations. Both furnish the *knowledge model* for the domain in question. Unfortunately, sometimes the terms have various

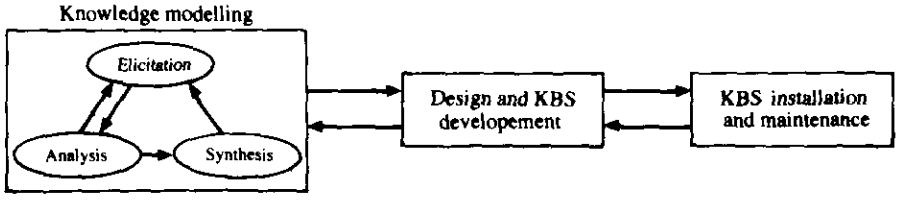


FIGURE 3. Phases in KBS development.

interpretations in the literature (Bradshaw & Boose, 1989; Ford *et al.*, 1993). However, though it is the case that they are sometimes used interchangeably in knowledge acquisition parlance, there is a difference between the two; they also play different roles in the knowledge modelling process. Since the distinction is useful, we will briefly define them.

A mediating representation aims to provide a medium of communication between the knowledge engineer(s) and a "grammar" of the expert's task (Johnson, 1989, page 184). It strives at conveying the sense of synthesis and coming to share some perspective with the expert of the domain through representations. Repertory grids which have been extensively used in knowledge acquisition (e.g. Shaw & Gaines, 1987) are a good example of a mediating representation. Another example is the concept map which also promotes communication and understanding via easily-learned generic knowledge representation forms. Hence, any representation which enhances communication amongst participants in the knowledge acquisition/modelling process and improves their understanding of evolving a conceptual domain model will pass as a mediating representation.

The literature is no precise on what intermediate representations are than they help bridge the wide gap between mediating representation and code. The intermediate representation would usually contain an integrated view of different knowledge types: concepts and concept definitions, causal relationships, functional dependencies, causal models, heuristics, list of previous cases, etc. However, intermediate representations must also be independent of any particular code-level formalism. These definitions will suffice for our hypothesis.

### 6.2. A PROPOSED FRAMEWORK

Figure 4 is a detailed elaboration of the first phase of Figure 3 which attempts to structure the notoriously difficult and *ad hoc* domain-modelling phase of knowledge based systems development. Note that Figure 3 highlights the three essential processes of elicitation, analysis and synthesis.

In so doing, is an attempt to provide a hypothesis for knowledge modelling as well as a principled basis for constructing more structured domain modelling/characterization tools. This hypothesis draws mainly from experiences of knowledge modelling exercises we have carried out ourselves using the SAAGS approach, but also from work of other researchers referenced in this paper. The data of our analyses are too voluminous to include here; as earlier mentioned, they typically amounted to 30–50 page documents. The lack of a systematic approach to knowledge modelling makes such an exercise even more crucial. Knowledge modelling is being done daily by knowledge engineers in an *ad hoc* fashion; hence, there is a pragmatic need for systemising the process. Culling from our experience, we propose the following underlying assumptions.

- During knowledge modelling the process of *elicitation* and *analysis* are tightly interwoven. Indeed a reviewer of this paper legitimately questioned if it is possible to analyse a domain and determine that it is hierarchical (for example) without eliciting a piece of information. We totally agree with the sentiments of this reviewer; however, it goes to show how intertwined the

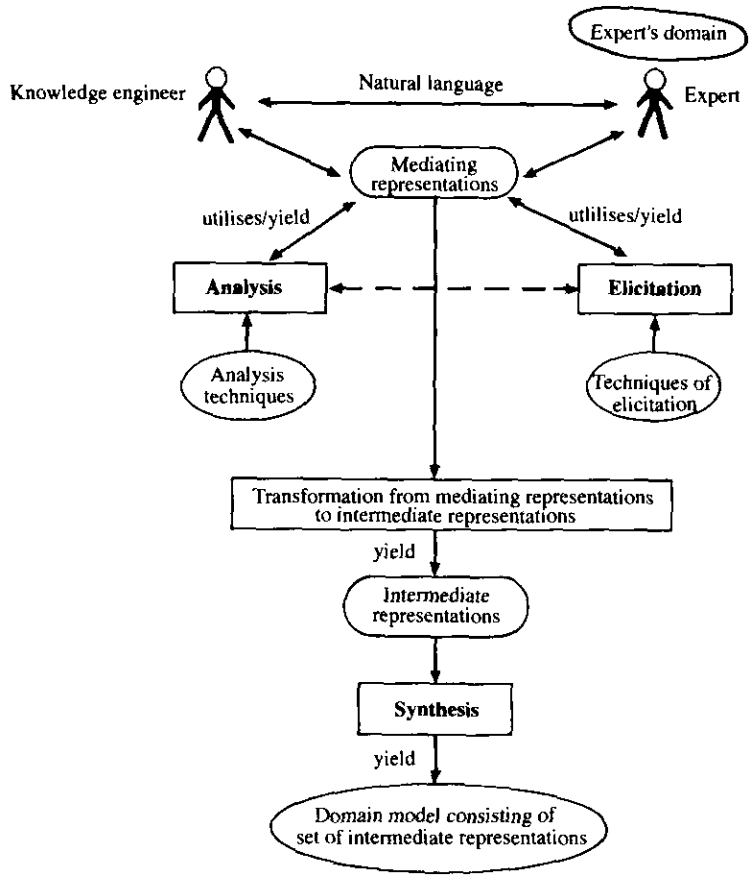


FIGURE 4. Structuring the knowledge modelling process.

processes are and most knowledge engineers acknowledge this. A central assumption of this proposal is that separating the two (artificial though it may be) is crucial to making knowledge modelling more explicit, transparent and principled.

- The product of knowledge modelling is a (set of) model(s) which basically comprise intermediate representations. A domain model is thus a *synthesis* of diverse intermediate representations.
- There exist distinct elicitation and analysis techniques, and these techniques yield distinct mediating representation knowledge types.
- Knowledge engineers and experts interact mainly through natural language with the aid of mediating representations.
- There exist mappings between mediating and intermediate representations and between the latter and code-level representations though we concede that these mappings are troublesome.
- Some mediating representations evolve into intermediate representations.

Drawing from these, the framework shown in Figure 4 results.

Essentially, a knowledge engineer is given a domain to model (normally, though not always, with the assistance of some expert) with a view to yielding a set of intermediate representations which together comprise the domain model. The knowledge engineer communicates with the expert in natural language and they come to share at least some perspective on the domain (Recogzei & Plantinga, 1987) with the aid of suitable mediating representations such as concept maps, hierarchies, etc. However, as any knowledge engineer will tell you the elicitation (interviewing) and analysis processes are closely intertwined. As Figure 4 illustrates, the elicitation and analysis processes have been "forced apart" in order to simplify the knowledge modelling process; hence, the rather weak link between them. There are several elicitation techniques in the literature which the knowledge engineer can use. Similarly, there are also domain analysis techniques such as that which we proposed earlier. We noted a domain can be characterized to include: a lexicon/glossary, a history, a theory, a metatheory, some basic metaphors, relations to other domains, a structure and its purpose.

A key point to note is that interacting with both the analysis and elicitation processes utilize and yield a variety of mediating representations. These mediating representations are later transformed into intermediate representations. Clearly, depending on the knowledge elicitation technique one uses, it utilizes and yields a different sort of mediating representation. For example, laddering will utilize/yield hierarchies (trees) while repertory grids yield facts and heuristics. However, analysis should have revealed *a priori* that the domain is hierarchical in nature before laddering is used; the idea here being that there must be a rationale for choosing the elicitation technique in the first place. Analysis, similarly, will utilize and yield different mediating representations depending on the top-level analytical primitive being considered. For example, the lexicon/glossary is just a list, while structure will yield diagrams, pictures, etc. The key point to emphasize here is all the representations being generated are domain-driven, *evolving either from considerations of the analysis and/or elicitation processes.* It is important to emphasize the modular nature of this hypothesis: for example, the analysis oval could be replaced with another set of techniques.

Often these various mediating representation knowledge types evolve/transform "naturally" into intermediate representations. However, sometimes they can map to several intermediate knowledge representations. In such a situation, we propose to have a library of intermediate representations from which a particular one may be chosen. This mapping process has been observed to be problematic. Indeed, when there is no obvious transformation, we resort to simply coercing the mediating representations into intermediate representations. The set of resulting intermediate representations are *synthesized* to provide the domain model. This latter process is itself non-trivial and may require the assistance of the expert. This is because problems of correspondence, conflict, etc., may arise.

Culling from our experiences and from the literature, mediating representations include concept maps, repertory grids, systemic grammar networks (SGNs), annotations, partially complete flow charts and diagrams, partially complete attribute and a-kind-of (AKO) lists, (concept) hierarchies, incomplete lists and tables, partial causal networks, equations, incomplete procedures and so forth.

Intermediate representations which we have produced have included structured

English (evolved from annotations), heuristics (elicited or evolved from repertory grids), flow charts, diagrams, task decomposition trees (evolved from hierarchies or repertory grids), domain/subdomain perspective diagrams showing the boundaries of the domain (evolved from concept maps), lists, tables, equations, pictures, glossary, event diagrams/charts, concept attribute lists, AKO lists, causal networks, annotations, etc. The domain models consist of these.

There are thus a plethora of mediating and intermediate knowledge representations. Clearly what is lacking is some information of *when* and *where* to use the various mediating and intermediate representations types, as well as guidelines on how the former transforms to the latter. A forthcoming paper (Nwana, in preparation) details various mediating and intermediate representations for knowledge modelling.

We concede to not having tested this framework/hypothesis, hence our current development of some tools.

## 7. Towards a tool for domain modelling

Section 6 has presented a hypothesis for domain modelling. Some of the section's contents may look trivial because it is what knowledge engineers do every day. However, this is the whole point. If it is not, why then is knowledge modelling so problematic? The lack of a systematic method to knowledge modelling leaves us in a situation where all we have is a bag of elicitation techniques and tools and with little guideance as to their usage. The fact is knowledge acquisition still remains much of an *art* which comes naturally to the gifted knowledge engineers. To be engineering, it needs to become more disciplined—less an art than a science, akin to the development of conventional software engineering techniques. This is necessary in order to train new knowledge engineers, to construct more maintainable knowledge based systems (KBSs), to facilitate their routine development, to improve their quality, etc. That is our key motivation for trying to structure the unstructured, i.e. striving to articulare what we believe comprises successful knowledge modelling which so far is undeniably an intractable problem despite the existence of numerous tools and techniques. We have also attempted to pull together the efforts of many researchers into what we believe is a plausible framework for knowledge modelling.

We also believe our proposal/hypothesis forms the basis for a hybrid tool for knowledge modelling. We have just commenced developing a suite of tools here at the University of Keele which strives to capture aspects of our hypothesis of Figure 4. The tool would also consist of various diagrammatic aids, facilities for structuring text, annotating and so forth. It will enable knowledge engineers to keep track of their actions when carrying out knowledge modelling, to document all that goes on at this vital phase of KBS development. We envisage the tool being used in both off-line and on-line mode depending on what cycle the knowledge engineer has reached in his iterative cycle. For example, initially, the early unstructured interviews may could be modelled using the tool off-line (i.e. expert not present in this context), while later structured sessions could use the tool on-line (i.e. expert present).

We also strive to keep our intermediate representations totally independent of all implementation concerns. In this way, we hope to avoid *shoe-horning* a domain into

some (set of) representations. We believe this to be one of the central problems in knowledge modelling: the fact that most knowledge acquisition tools/techniques are driven by implementation concerns rather than by the nature of the domain (Shaw, 1989; Woodward, 1990). The fact that we have carried out several domain modelling exercises *without* a view to developing KBSs did help us in this goal. It actually emerged that some of the industrial domains we modelled were not feasible/suitable for KBS development thereby saving the costs of a possible failed project.

There also appears to exist a continuum of representations ranging from mediating representations at one end via intermediate representations in the middle to machine representations at the other end. We hope to construct tools to aid such transformations so as to be able to eventually generate code. For example, consider the following:

concept maps → perspective diagrams → semantic networks → conceptual graphs → predicate calculus → PROLOG
concept maps → concept-relationship diagrams → semantic networks → conceptual graphs → predicate calculus
concept maps → event diagrams → temporal semantic nets/objects
Incomplete flow charts → Complete flow charts → Procedures (Algorithms)
repertory grids → facts → knowledge base facts
repertory grids → heuristics → knowledge base rules
Incomplete attribute lists → Concept attribute lists → Objects/Frames
Incomplete AKO lists → Concept AKO lists → Objects/Instances/Inheritance hierarchies
concept maps → causal networks → inference networks → rules
Incomplete task hierarchies → Complete task hierarchies → generic tasks models

In summary, we envisage the proposed tool to characterize/model domains will go some way in addressing some of the key problems with knowledge Modelling which we restated in section 4.

## Conclusions

This contribution of this paper can be considered to be four-fold. Firstly, it lists some problems which are critical to knowledge modelling. Secondly, it critically reviews current techniques and tools in order to establish the case that they do not address (but circumvent) these real problems. Thirdly, it suggests that knowledge modelling should be driven by the nature of the domain being modelled, else knowledge imposition rather than knowledge acquisition results. Lastly, it proposes a framework for knowledge modelling.

This research was sponsored by Shell Research UK (Thornton) and Unilever Research (Port Sunlight). The authors sincerely acknowledge the comments of Drs Marc Linster (then of GMD, St Augustin) and Anjo Anjewierden (University of Amsterdam) on an earlier draft of part of this paper. Discussions with Enrico Motta (Open University, UK) have also proved very helpful. We acknowledge especially the encouragements from Drs Ken Lunn, Keith McFarlane and Hugh Dorans of Shell Research UK. We also acknowledge the detailed comments of Professor Brian Gaines of the University of Calgary and another anonymous

referee which has greatly improved the quality of the paper. The comments of many anonymous EKAW'91 and EKAW'92 referees are also acknowledged.

## References

ALEXANDER, J. H., FREILING, M. J., SHULMAN, S. J., REHFUS, S. & MESSICK, S. L. (1987). Ontological analysis: an on going experiment. *International Journal of Man–Machine Studies* **26,** 473–485.

ANJEWIERDEN, A. (1987). Knowledge Acquisition Tools. *AICOM* **0,** 29–38.

ANJEWIERDEN, A., WIELEMAKER, J. & TOUSSAINT, C. (1992). Shelley–computer-aided knowledge engineering. *Knowledge Acquisition* **4,** 109–125.

BENCH-CAPON, T. J. M., COENEN, F., NWANA, H. S., PATON, R. C. & SHAVE, M. J. R. (1993). Two Aspects of the Validation and Verification of Knowledge Based Systems. *IEEE Expert,* 76–81.

BENNETT, J. S. (1985). A knowledge-based reasoning system for acquiring the conceptual structure of a diagnostic expert system. *Journal of Automated Reasoning* **1,** 49–74.

BOOSE, J. H. & BRADSHAW, J. M. (1987). Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for expert systems. *International Journal of Man–Machine Studies* **26,** 3–28.

BOOSE, J. H. (1985). A knowledge acquisition program for experts systems based on personal construct psychology. *International Journal of Man–Machine Studies* **23,** 495–525.

BOOSE, J. H. (1989). A survey of knowledge acquisition techniques and tools. *Knowledge Acquisition* **1,** 3–37.

BRADSHAW, J. & WOODWARD, B. (1989). Knowledge Acquisition Tools. (Panel Discussion Summary of IJCAI-89 Knowledge Acquisition Workshop), In BOOSE, J. & GAINES, B. (Eds), *Knowledge Acquisition for KBS Newsletter* **1**(3), 12.

BRADSHAW, J. M. & BOOSE, J. H. (1989). Knowledge Acquisition as CASE for Knowledge Based Systems. *Paper presented at the 3rd International Workshop on Computer-Aided Software Engineering (CASE-89),* London, July.

BREUKER, J. A., WIELINGA, B. J., VAN SOMEREN, M., DE HOOG, R., SCHREIBER, G., DE GREEF, P., BREDEWEG, B., WIELMAKER, J., BILLEAUT, J. P., DAVOODI, M. & HAYWARD, S. (1987). *Model-Driven Knowledge Acquisition Interpretation Models.* Deliverable Task A1, ESPRIT Project 1098, Commission of the European Community.

BUCHANAN, B. G., BARSTOW, D., BECHTAL, R., BENNETT, J., CLANCEY, W. J., KULUKOWSKI, C., MITCHELL, T. & WATERMAN, D. A. (1983). Constructing an Expert System. In HAYES-ROTH, F., WATERMAN, D. A. & LENAT, D. B. (Eds), *Building Expert Systems,* Reading, MA: Addison-Wesley.

BURTON, A. M., SHADBOLT, N. R., HEDGECOCK, A. P. & RUGG, G. (1987). A formal evaluation of elicitation techniques for expert systems: domain 1. In MORALEE, D. S. (Ed), *Research and Developments in Expert Systems,* Cambridge: Cambridge University Press.

CHANDRASEKARAN, B. (1985). Generic tasks in knowledge based reasoning: characterizing and designing expert systems at the "right" level of abstraction. *Proceedings of the IEEE 2nd Conference on Artificial Intelligence Applications,* 294–300.

CHANDRASEKARAN, B. (1989). Task structures, knowledge acquisition and learning. *Machine Learning* **4,** 339–345.

CLANCEY, W. J. (1989). The knowledge level reinterpreted: modeling how systems interact. *Machine Learning* **4,** 285–291.

CLANCEY, W. J. (1985). Review of J. F. Sowa's conceptual structures. *Artificial Intelligence* **27,** 113–128.

CLANCEY, W. J. (1986). Heuristic Classification. In KOWALIK, J. S. (Ed), *Knowledge based Problem Solving,* Englewood Cliffs, NJ: Prentice Hall, 1–67.

CLANCEY, W. J. (1993). The knowledge level reinterpreted: modelling socio-technical systems. *International Journal of Intelligent Systems* **8,** 33–49.

DAVIS, R. & LENAT, D. B. (1982). *Knowledge-Based Systems in Artificial Intelligence,* New York: McGraw-Hill.

DAVIS, R. (1979). Interactive transfer of expertise: acquisition of new inference rules. *Artificial Intelligence* **12,** 121–157.

DIEDERICH, J., RUHMANN, I. & MAY, M. (1987). KRITON: a knowledge acquisition tool for expert systems. *International Journal of Man–Machine Studies* **26,** 29–40.

DOWNS, E., CLARE, P. & COE, I. (1988). *Structural Systems Analysis and Design Method: Application and Context,* Hemel Hempstead: Prentice Hall.

ESHELMAN, L., EHRET, D., MCDERMOTT, J. & TAN, M. (1987). MOLE: a tenacious knowledge acquisition tool. *International Journal of Man–Machine Studies* **26,** 41–54.

FORD, K. M., BRADSHAW, J. M., ADAMS-WEBBER, J. &. AGNEW, N. M. (1993). Knowledge acquisition as a constructive modelling activity. *International Journal of Intelligent Systems* **8**(1), 9–32.

FRIEDLAND, P. E. & IWASAKI, Y. (1985). The concept and implementation of skeletal plans. *Journal of Automated Reasoning* **1,** 161–208.

GAINES, B. R. (1987). An overview of knowledge acquisition and transfer. *International Journal of Man–Machine Studies* **26,** 183–202.

GAINES, B. R. (1994). Personal communication.

GROVER, M. D. (1983). A pragmatic knowledge acquisition methodology. *Proceedings of IJCAI* **8,** 436–438.

GRUBER, T. (1988). Acquiring strategic knowledge from experts. *International Journal of Man–Machine Studies* **29,** 579–597.

GUIDA, G. & TASSO, C. (1989). Building expert systems: from life cycle to development methodology. In GUIDA, G. & TASSO, C. (Eds), *Topics in Expert Systems Design, Methodologies and Tools,* Amsterdam: North Holland, 3–24.

HAYWARD, S. A., WIELINGA, B. J. & BREUKER, J. A. (1987). Structured analysis of knowledge. *International Journal of Man–Machine Studies* **26,** 453–472.

JOHNSON, N. E. (1989). Mediating representations in knowledge elicitation. In DIAPER, D. (Ed), *Knowledge Elicitation: Principles, Techniques and Applications,* Chichester: Ellis Horwood, 179–194.

KAHN, G., NOWLAND, S. MCDERMOTT, J. (1985). MORE: an intelligent knowledge acquisition tool. In *Proceedings of the International Joint Conference on Artificial Intelligence,* Los Angeles, CA, 581–584.

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

KITTO, C. M. & BOOSE, J. H. (1989). Selecting knowledge acquisition tools and strategies based on application characteristics. *International Journal of Man–Machine Studies* **31,** 149–160.

MARCUS, S. (1988). SALT: a knowledge acquisition tool for propose-and-refine systems. In MARCUS, S. (Ed), *Automating knowledge acquisition for expert systems,* Boston: Kluwer Academic Publishers.

MARTIN, J. & OXMANN, S. (1988). *Building Expert Systems: A Tutorial,* Englewood Cliffs, NJ: Prentice-Hall.

MCDERMOTT, J. (1986). Making expert systems explicit. *Proceedings of IFIP World Congress.* Dublin.

MCDERMOTT, J. (1988). Preliminary steps towards a taxonomy of problem-solving methods. In MARCUS, S. (Ed), *Automating knowledge acquisition for knowledge based systems.* Chapter 8, 120–146.

MICHALSKI, R. S. (1983). Theory and methodology of inductive learning. In MICHALSKI, R. S., CARBONELL, J. G. & MITCHELL, T. M. (Eds), *Machine Learning: An Artificial Intelligence Approach.* New York: Tioga.

MITCHELL, T. M., MAHADEVAN, S. & STEINBERG, L. I. (1985). LEAP: a learning apprentice for VLSI design. *Proceedings of the International Conference on Artificial Intelligence,* Los Angeles.

MORIK, K. (1989). Sloopy Modelling. In MORIK, K. (Ed), *Knowledge Representation and Organisation in Machine Learning: Lecture Notes in Artificial Intelligence* Vol. 347. London: Springer Verlag. 107–134.

MORIK, K. (1991). Underlying Assumptions of Knowledge Acquisition and Machine Learning. *Knowledge Acquisition* **3,** 137–156.

MORIK, K. (1993). Balanced cooperative modelling. *Machine Learning* **11,** 217–235.

MOTTA, E., EISENSTADT, M., PITMAN, K. & WEST, M. (1988). Knowledge acquisiton in KEATS: The knowledge engineer's assistant. *Expert Systems* **5**(2).

MOTTA, E., RAJAN, T. & EISENSTADT, M. (1990). Knowledge Acquisition as a process of model refinement. *Knowledge Acquisition* **2,** 21–49.

MUSEN, M. A. (1989). Conceptual models of interactive knowledge acquisition tools. *Knowledge Acquisition* **1,** 73–88.

MUSEN, M. A., FAGAN, L. M., COMBS, D. M. & SHORTLIFFE, E. H. (1987). Use of a domain model to drive an interactive knowledge-editing tool. *International Journal of Man–Machine Studies* **26,** 105–121.

NEWELL, A. (1982). The knowledge level. *Artificial Intelligence* **18,** 87–127.

NWANA, H. S. (in preparation). Mediating and Intermediate Representations for Knowledge Acquisition.

NWANA, H. S., PATON, R. C., SHAVE, M. J. R. & BENCH-CAPON, T. J. M. (1991). Textual Analysis for Knowledge Acquisition using the MEKAS Approach. *Proceedings of the 5th European Knowledge Acquisition Workship (EKAW-91),* Crieff, Scotland, May.

O'NEILL, M. & MORRIS, A. (1989). Expert systems in the United Kingdom: an evaluation of development methodologies. *Expert Systems* **6,** 90–98.

PARSAYE, K. & CHIGNELL, M. (1988). *Expert Systems for Experts.* Toronto: Wiley.

PATON, R. C., NWANA, H. S., SHAVE, M. J. R., BENCH-CAPON, T. J. M. & HUGHES, S. (1991). Foundations of a Structured Approach to Characterising Domain Knowledge. *Cognitive Systems* **3,** 139–161, November.

PATON, R. C., NAWAN, H. S., SHAVE, M. J. R., BENCH-CAPON, T. J. M. (1991). From real world problems to domain characterisations. *Proceedings of the 5th European Knowledge Acquisition Workshop (EKAW-91),* Crieff, Scotland, May.

QUINLAN, J. R. (1983). Learning efficient classification procedures and their applications to chess and games, In MICHALSKI, R. S., CARBONELL, J. G. & MITCHELL, T. M. (Eds), *Machine Learning: An Artificial Intelligence Approach.* New York: Tioga.

RAPPAPORT, A. T. & GAINES, B. R. (1988). Integrating knowledge acquisition and performance systems. *Proceedings of AAAI-88 Workshop on Integration of Knowledge Acquisition and Performance Systems,* St Paul, MN.

RECOGZEI, S. & PLANTINGA, E. P. O. (1987). Creating the domain of discourse: ontology and inventory. *International Journal of Man–Machine Studies* **27,** 235–251.

SANDBERG, J. (1991). Interview with Bill Clancey, *AI Communications.*

SCHREIBER, G., BREUKER, J., BREDEWEG, B. & WIELINGA, B. (1988). Modelling in KBS Development. *Proceedings of Third European Workshop on Knowledge Acquisition for Knowledge Based Systems, EKAW '88 (June),* Bonn.

SHAW, M. L. G. & GAINES, B. R. (1987). Techniques for knowledge acquisition and transfer. *International Journal of Man–Machine Studies* **27,** 251–280.

SHAW, M. L. G. & GAINES, B. R. (1988). A methodology for recognising consensus, correspondence, conflict and contest in a knowledge acquisition system. *Proceedings of Workshop on Knowledge Acquisition for Knowledge Based Systems.* November, Banff, Canada.

SHAW, M. L. G. (1989). A grid-based tool for knowledge acquisition. *Proceedings of IJCAI-1989 Workshop on Knowledge Acquisition,* Detroit, Michigan, 19–22.

SHAW, M. L. G. & WOODWARD, J. B. (1990). Modelling expert knowledge. *Knowledge Acquisition* **2,** 179–206.

SHORTLIFFE, E. H., SCOTT, A. C., BISCHOFF, M. B., VAN MELLE, W. & JACOBS, C. D. (1981). ONCOCIN: an expert system for oncology protocol mangement. In *Proceedings of the 7th International Joint Coinference on Artificial Intelligence,* Vanouver, Canada, 876–881.

STEELS, L. (1990). Components of Expertise. *AI Magazine,* Summer 1990, 20–49.

VAN MELLE, W. (1979). A domain-independent production rule system for consultation programs. *Proceedings of the International Joint Conference on Artificial Intelligence.*

WATERMAN, D. (1986). *A Guide to Expert Systems,* Reading, MA: Addison Wesley.

WELBANK, M. (1990). A overview of knowledge acquisition methods. *Interacting with Computers* **2,** 83–91.

WIELINGA, B. J., SCHREIBER, A. & BREUKER, J. A. (1992). KADS: a modelling approach to knowledge engineering. *Knowledge Acquisition* **4,** 5–53.

WOODWARD, B. (1989). General Structures as Domain Models to Guide Knowledge Acquisition. *Proceedings of IJCAI-1989 Workshop on Knowledge Acquisition,* Detroit, Michigan, 10–14.

WOODWARD, B. (1990). Knowledge acquisition at the front end: defining the domain. *Knowledge Acquisition* **2,** 73–94.