KNOWLEDGE MODELING FOR SOFTWARE DESIGN

Mildred L G Shaw & Brian R Gaines

Knowledge Science Institute
University of Calgary
Calgary, Alberta, Canada T2N 1N4
mildred@cpsc.ucalgary.ca, gaines@cpsc.ucalgary.ca

Abstract: This paper develops a modeling framework for systems engineering that encompasses systems modeling, task modeling, and knowledge modeling, and allows knowledge engineering and software engineering to be seen as part of a unified developmental process. This framework is used to evaluate what novel contributions the 'knowledge engineering' paradigm has made, and how these impact software engineering.

INTRODUCTION

In the knowledge acquisition community the development of tools for eliciting knowledge from experts has come to be seen as a 'knowledge modeling' exercise in which human practical knowledge is modeled within the computer (Gaines, Shaw and Woodward, 1992). It has been suggested that a common factor underlying all knowledge-based systems, including software design systems, is that they contain qualitative world models, and that we can gain insights into the structure of knowledge bases and knowledge engineering by classifying the types of models involved (Clancey, 1989). These considerations suggest that a classification of the sources and types of models developed in system engineering may be used to provide a framework within which knowledge engineering and software engineering methodologies and tools can be analyzed and compared.

One might view the replication of human expertise in a knowledge-based system as involving the elicitation of the *mental models* of the human experts involved (Gentner and Stevens, 1983). However, we do not have direct access to these models, and must create *conceptual models* of them through communication with the expert (Norman, 1983). The representations made by the knowledge engineer are not isomorphic to structures in the mind of the expert (Compton and Jansen, 1990). Within this framework, one can view knowledge engineers, or automated knowledge acquisition systems interacting with the expert, as accessing and developing the expert's conceptual models. Some parts of these models may be pre-existent, particularly if the expert has a teaching role, but other parts will come into being as a result of the knowledge acquisition process.

The distinction that Norman introduces between mental models and conceptual models, and the dubious status of mental models in themselves, suggests that a useful framework for the analysis of knowledge engineering may be developed through the analysis of the sources and types of conceptual model available to the knowledge engineer rather than focusing only on the mental processes underlying expertise. The situation of the introspective expert who can communicate his or her 'knowledge' well, may be treated as one where the 'knowledge engineering' and 'expert' roles are operating effectively together within the same person. The situation of the expert from whom knowledge is being 'elicited' actually building a new model on the basis of his or her skills through the process of elicitation may be treated as one where the conceptual model is developed as part of the process of knowledge engineering. In adopting the conceptual modeling perspective we do not exclude previous viewpoints, but rather supplement them with complementary perspectives.

A MODELING FRAMEWORK FOR INFORMATION SYSTEM DEVELOPMENT

It is customary in expert system development, to assume that the expert has already constructed such models or may be in a privileged position to do so through self-observation and introspection, and these may be elicited by direct communication between knowledge engineer and expert. Additionally, the knowledge engineer may derive models from other experts, from the literature, and from the application of principles allowing performance skills to be derived from deep knowledge. The final knowledge-based system development involves the synthesis of these many models and the encoding of them to become an operational knowledge-based systems emulating the desired expertise.

Thus, the knowledge engineer, or knowledge engineering team and tools, has access to multiple sources of data through various channels and uses these to develop a variety of conceptual models. Figure 1 shows the major conceptual models that may be developed in knowledge engineering, distinguished by their sources, and indicating some of the knowledge engineering processes and skills involved. This figure attempts to be comprehensive, showing knowledge sources not only in association with the expert and his or her behavior, but also knowledge derived from others, the literature and through the application of laws and principles.

Figure 1 is an accurate representation of what is typically involved in knowledge engineering for a knowledge based system development nowadays. It uses any source of knowledge that is available for system development, not just the practical reasoning of the expert, and hence exemplifies the "second type" of knowledge engineering cited above (Feigenbaum, McCorduck and Nii, 1988). However, it still has a major, and irreducible component of the first type representing the central expert systems paradigm. What is significant is the way in which the two approaches are synthesized, and also the way in which many components of the "second type" of activity are already part of modern systems and software engineering. This is the basis of a much wider synthesis than that between two forms of knowledge engineering.

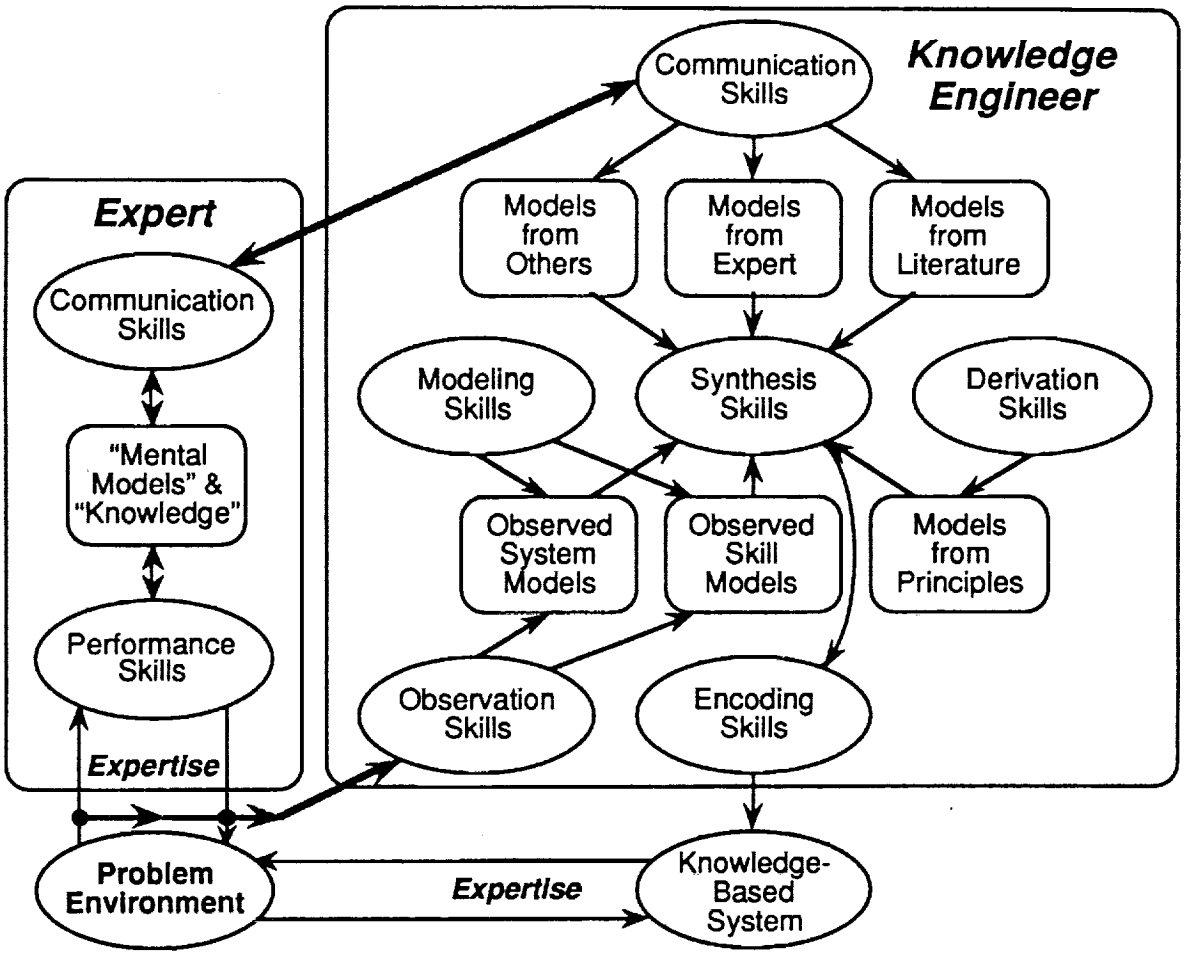

Fig.1 Modeling processes in knowledge engineering

A MODELING FRAMEWORK FOR INFORMATION SYSTEM DEVELOPMENT

The discussion of the preceding sections and the range of modeling processes shown in Figure 1 provide an overall framework for systems engineering in terms of the sources and types of models involved. Within such a framework it should become only a matter of internal classification and terminology that a method is part of a 'knowledge engineering' or a 'software engineering' approach, rather than a resultant system classification.

Figure 2 presents a modeling framework for knowledge acquisition methodologies, techniques and tools based on the distinctions already discussed and the incorporation of system analysis and software engineering procedures. In the leftmost column are the knowledge sources in terms of systems and modeling schema already discussed with the addition, at the top, of 'objective models' as a term for the formally specified operational models. In the column to the right of this are the processes giving access to these models. These processes are shown as mediating between the systems and models involved, deriving from and generating, the hierarchical relation between the systems and models in the leftmost column.

In the next column on the right are shown the knowledge acquisition procedures appropriate to each of the access processes. These generate data and knowledge bases as shown to their right, which are in one-to-one correspondence with the original systems and models in the leftmost column. In the rightmost column are shown analysis and synthesis techniques that draw on these databases to generate the computational knowledge base, and also mediate between them generating one form of data or knowledge from another. These combine with synthesis techniques that integrate the results of analysis and of derivations from various knowledge sources to synthesize a computational knowledge base.

Thus the overall schema consists of five types of component:

1. *Systems and modeling schema*: the problem environment, performance skill to be emulated, expert's mental models, knowledge engineer's conceptual models, and, possibly, objective models.
2. *Access processes:* instrumentation of the target system, the expert's interaction with it, his or her introspection about the skill, communication about it, and its expression in formal terms as objective knowledge.

3. *Knowledge acquisition procedures*: observation of the target system, observation of the expert's behavior, elicitation procedures, discourse procedures, formalization procedures, and implementation procedures.
4. *Data and knowledge bases:* database of system data; database of behavioral data; informal knowledge base; formal knowledge base; computational knowledge base; objective models.
5. *Analysis and synthesis procedures*: classical system identification can be used to build system models from observation data; empirical induction and case-based clustering can be used to build skill models from behavioral data; conceptual organization and linguistic analysis techniques can be used to build a formal, or structured, knowledge base from an informal, or intermediate, one; knowledge modeling techniques can be used to represent the formal knowledge base in computational form; and logical deduction from laws and principles may be used to provide some knowledge about a system and this, together with the results of data analyses from various sources needs to be integrated to form a computational knowledge base.

All the earlier stages of analysis are shown as normally creating data at the next level but also as potentially creating computational systems in their own right.

Figure 2 illustrates the way in which knowledge engineering as a system design methodology is sandwiched between two classical approaches to system engineering. At the bottom is the path to system design through instrumentation, data collection and system identification. At the top is the path to system design through existing objective knowledge of the physical world allowing explication of particular requirements to lead directly to implementation. The middle layers represent the enrichment of the design process when we draw on human skills as exemplars of the system to be designed. Such a process has been common informally in engineering design, and knowledge engineering may be seen as formalizing it now that computer technology makes it feasible to develop knowledge-based systems operationalizing human expertise.

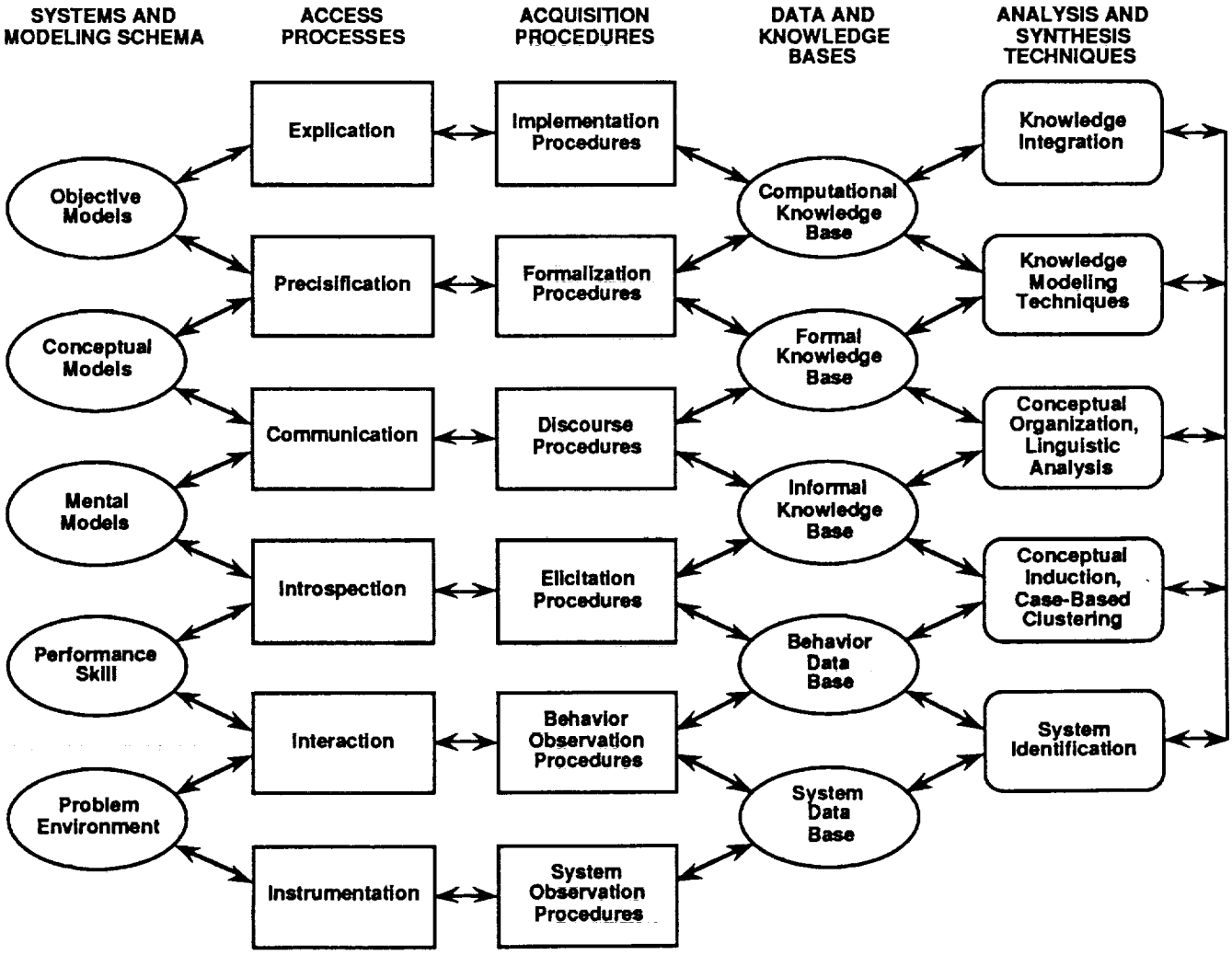

Fig.2 A hierarchical framework for knowledge acquisition

KNOWLEDGE ACQUISITION ISSUES IN TERMS OF THE FRAMEWORK

It is clear that a catchall term such as 'interviewing' does not designate a monolithic technique in terms of the framework of Figure 2. When we interview an expert we may be operating at any level of the hierarchy and may be supporting any one of the many processes shown. All that we can say about interviewing in general is that a flow of linguistic information is involved—it is the content of that flow that determines the type of knowledge engineering involved. The expert may provide observations of the system, observations of his or her own problem solving behavior, introspection about aspects of his or her mental models, statements about his or her conceptual models of any aspect of the situation, and statements of formal or even computational models relating to the situation.

Specific knowledge acquisition techniques are characterized by their vertical and horizontal locations within the framework. For example, protocol analysis involves data collection for the behavior data base through observation of interaction at one level or elicitation of introspection at the next. The behavior database is then subject to statistical system identification or to conceptual induction and clustering. The data collection methodology in protocol analysis may easily slip into the elicitation of not just a protocol but also an explanatory commentary which belongs in the informal knowledge base and is subject to linguistic analysis. Thus, applications of protocol analysis may involve multiple levels and activities that are confusing unless seen as organized within the framework.

Analytical tools such as induction and clustering algorithms have a well-defined location in the framework as analysis techniques providing a model creation technology. Their differentiation comes from what level, or levels, they can accept data, and at what level, or levels, they create data. A major focus in machine learning research for several years has been to create models at the knowledge level, conceptual structures rather than rules. To the extent that all the analytic techniques involved do this, the problem becomes one of integration of conceptual structures. However, it is more usual to find that the analytic tools create data or knowledge at different levels and further processing is required before integration is possible.

Methodologies such as KADS (Akkermans, Harmelen, Shreiber and Wielinga, 1992) that provide a structured software engineering approach to knowledge engineering are focused at the penultimate level of applying formalization procedures to derive a formal knowledge base through making conceptual models precise. KADS focuses on the detailed structure of a formal problem solving architecture within which to operationalize the results of knowledge acquisition rather than on the processes of knowledge acquisition themselves. It may be seen as providing a formally specified 'virtual machine' well-suited to the range of system developments that have come to be classified under the heading of 'knowledge-based systems.' Less formally, one can say that it provides a 'high-level language' in contrast to the 'machine languages' provided by expert system shells.

Knowledge acquisition methodologies such as those stemming from personal construct psychology (Shaw, 1980) that are based on a cognitive model of intelligent agents are focused on the middle levels in Figure 2, modeling the way in which mental models mediate between conceptual models and performance skills. Clearly any well-founded cognitive psychology has a potential role to play in knowledge acquisition that is strictly within the 'expert systems' paradigm of modeling the expert rather than the system. However, to be useful the psychology must result in operational models on the one hand and support methodologies giving access to its hidden variables on the other. Personal construct psychology has been particularly attractive in these respects because, even though it is a constructivist model, it takes a positivist, axiomatic approach based on a few well-defined primitives that correspond to a formal intensional logic (Gaines and Shaw, 1992), and is well-supported by practical tools (Boose and Bradshaw, 1987; Shaw and Gaines, 1987; Shaw and Gaines, 1989).

The interface between cognition and formalization for people is mediated through language and knowledge acquisition support is required for the communication and discourse procedures and analysis level in Figure 2. Current knowledge acquisition tools addressing this level range from those focusing on the inter-translation of restricted natural language and knowledge representation frames such as SNOWY (Gomez and Segami, 1990), to those providing support for human classification of natural language components in terms of knowledge level primitives such as Cognosys (Woodward, 1990). Improved natural language processing must have a very high priority in the support of the complete range of knowledge acquisition processes in the framework of Figure 2.

Classical system analysis focuses on the collection and analysis of system and behavior data at the lower levels of Figure 2. In complex system development the other levels play their part, but the basic assumption has been that the final system design is grounded in accurate models of the environment in which the system is to operate and in precise 'requirements specifications' corresponding to the top level goals of the human agents involved. The implementation is quite separate from the system analysis and design because conventional programming languages do not provide knowledge-level constructs supporting human understanding of their operation.

In this respect, the framework of Figure 2 may be seen as an extension to classical system analysis appropriate to knowledge-based systems where very high level languages at the 'knowledge level' are being used for the implementation to provide this support of human understanding.

CONCLUSIONS

A complete account of system engineering acquisition for modern advanced information systems requires the integration of classical system analysis, cognitive modeling of intelligent agents, linguistic analysis of text and discourse, and a rich formal language at the knowledge level. This integration would provide us with a system development methodology adequate to cope with the increased expectations of those specifying requirements for knowledge-based systems.

However, note that the knowledge level language alone is only a target for specification. On the one hand it needs to be made operational as computational knowledge. On the other it needs to maintain an effective ongoing relation with the knowledge processes that drive it, many of which are those of active human agents forming an essential component of the ongoing system operation. Knowledge acquisition should not be seen as part of the system design process only. Knowledge is dynamic and changing, and acquisition, maintenance and upgrading must merge into one process that is fully supported as an ongoing system operation. In particular, the cognitive aspects of much of the knowledge must continue to be recognized and supported in the ongoing system operation. Formalization cannot be at the expense of human understanding. On the contrary, effective formalization should lead to enhanced human understanding. This is the greatest challenge in the development of an effective knowledge-based systems technology. The objective is not just emulation of isolated human peak performance, but rather the emulation of the total human ability to develop, adapt and maintain that performance in a dynamic and uncertain environment.

ACKNOWLEDGEMENTS

This work was funded in part by the Natural Sciences and Engineering Research Council of Canada. We are grateful to our colleague Brian Woodward for discussions that have influenced this paper.

REFERENCES

Akkermans, H., Harmelen, F. van, Shreiber, G. and Wielinga, B. (1992). A formalisation of knowledge-level models for knowledge acquisition. **International Journal of Intelligent Systems** to appear.

Boose, J.H. and Bradshaw, J.M. (1987). Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for knowledge-based systems. **International Journal of Man-Machine Studies 26** 3-28.

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

Compton, P. and Jansen, R. (1990). A philosophical basis for knowledge acquisition. **Knowledge Acquisition** 2(3) 241-258.

Feigenbaum, E., McCorduck, P. and Nii, H.P. (1988). **The Rise of the Expert Company**. New York, Times Books.

Gaines, B.R. and Shaw, M.L.G. (1992). Basing knowledge acquisition tools in personal construct psychology. **Knowledge Engineering Review** to appear.

Gaines, B.R., Shaw, M.L.G. and Woodward, J.B. (1992). Modeling as a framework for knowledge acquisition methodologies and tools. **International Journal of Intelligent Systems** to appear.

Gentner, D. and Stevens, A., Ed. (1983). **Mental Models**. Hillsdale, New Jersey, Erlbaum.

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

Norman, D.A. (1983). Some observations on mental models. **Mental Models**. Hillsdale, New Jersey, Erlbaum.

Shaw, M.L.G. (1980). **On Becoming A Personal Scientist: Interactive Computer Elicitation of Personal Models Of The World**. London, Academic Press.

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

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

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