EXPERIENCE WITH A CLASS LIBRARY FOR KNOWLEDGE REPRESENTATION AND COMMUNICATION

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

Abstract: The implementation architecture of a knowledge representation server as a class library in an object-oriented language is described. The graphic editing tools developed to support ease of interactive editing of the objects in the knowledge representation system are illustrated. It is shown that object-oriented paradigms are effective at all levels of system development—software engineering of high-performance modules, intelligent communication between heterogeneous subsystems, and knowledge representation for organizational modeling. They allow system architectures to be developed which combine centralized repositories and distributed heterogeneous sub-systems to support new forms of organizational integration required in computer-integrated manufacturing.

INTRODUCTION

Manufacturing organizations are complex structures of human roles and relations, information, financial and material flows, power and accountability relations, skills and knowledge (Scott, 1981). However, these structures have been only implicitly represented in current information technology. This is not by choice, but because our 'high level' programming languages, while a major advance on machine code, have focused on efficient data processing rather than organizational modeling (Scheer, 1989). Normalized relational, and entity-relation, databases have some degree of corporate modeling capability, but their limited passive representations may be seen as capturing data about the activities of an organization rather than modeling its procedures, knowledge and skills (Gaines, 1988a). It is only in recent years that the combination of advances in computing power and software engineering techniques has allowed practical application of object-oriented paradigms in which modeling and data processing can be effectively combined (Gupta, 1991).

In terms of overall architecture, one can see a convergence between centralized corporate knowledge repositories as extensions of current databases (Davis, 1989), and distributed, heterogeneously integrated knowledge bases as extensions of networked personal computing environments. A combination of both is required since computational complexity limitations (Nebel, 1990) in deductive databases severely limit the scalability of current expert systems to become 'corporate minds', yet the anarchy of uncontrolled distribution would lead to precisely those problems of inconsistency and unreliability of data that the repository approach seeks to avoid (Sagawa, 1990).

As part of such a combined architecture, knowledge representation and reasoning systems have an important role to play in the overall integration of manufacturing systems. They provide facilities for systems modeling and integration at the knowledge level that support both designers and managers in developing and using systems effectively. However, to play this role effectively, the knowledge-based components must integrate well with other applications, and offer light-weight, portable and embeddable services. These are software engineering considerations best currently addressed by object-oriented paradigms.

The approach taken in the studies reported here has been to implement recent advances in the design of well-specified knowledge representation and reasoning systems as a light-weight knowledge representation server, and to use the object-oriented programming paradigm to surround the server with small functional modules supporting a variety of applications (Gaines, 1991a,b). The server is sufficiently light-weight that multiple instantiations of it can exist on personal workstations, yet sufficiently fast and powerful to cope with knowledge structures involving thousands of concepts and tens of thousands of entities.

The approach taken to data and knowledge presentation, retrieval, inference and editing has been to develop a formal visual language supported through graphic editors providing a visualization and drawing environment similar to that of drawing tools and with strong attention to principles of cognitive ergonomics (Gaines, 1991c). More conventional, user interfaces to spreadsheets, databases, word-processing and accounting packages are also provided largely for factual material, but the primary interface to the knowledge structures representing conceptual material is through the visual language. It supports visual libraries of conceptual and problem-solving structures that can be linked with specifically developed knowledge structures, so that the balance between centrally and locally managed knowledge structures can be varied and managed as required.

This paper described the design and implementation of a knowledge representation server with graphic language interfaces as a class library in C++.

KSSn ARCHITECTURE

KSSn (Knowledge Support System n) is the latest in a series of developments deriving from our initial implementation of KSS0 (Gaines, 1988b) and KSS1 (Shaw & Gaines, 1987). These early knowledge support systems focused on knowledge acquisition and conceptual modeling, and have been extended through heterogeneous integration to offer close integration with hypermedia and expert system shells (Gaines & Linster, 1990; Gaines, Rappaport & Shaw, 1992). One problem with research and applications in this area is that of rapid development in techniques, tools and technologies. Systems become obsolescent in a very short time and a monolithic architecture is difficult to maintain and enhance. To cope with this we have completely reengineered KSSn as a C++ class library implementing KRS, a KL-ONE-like knowledge representation server, and a set of associated functional modules for knowledge elicitation, text analysis, empirical induction, graphic knowledge base editing, and so on.

This approach is based on two recent developments that open up new possibilities for meaningful empirical research in artificial intelligence: new insights into the complexity issues affecting knowledge representation and reasoning (Brachman & Levesque, 1984; Schmidt-Schauss, 1989); and the development of object-oriented programming environments offering precise definition and efficient implementation of extensible abstract data types (Shriver & Wegner, 1987). The advances in artificial intelligence allow requirements specifications to be developed for knowledge representation and reasoning systems with known, and practically applicable, performance curves (Borgida, Brachman, McGuinness & Resnick, 1989). The advances in software engineering allow these specifications to be implemented cleanly in such a way that their conformance with the requirements can be verified and their algorithms achieve the best possible performance.

The series of knowledge representation schemes and tools commencing with KL-ONE (Brachman & Schmolze, 1985), developing through KRYPTON (Brachman, Gilbert & Levesque, 1985) and currently culminating in systems such as CLASSIC (Borgida, Brachman, McGuinness & Resnick, 1989) and LOOM (MacGregor, 1988) has reached a maturity of technology which offers the promise of knowledge representation 'services' in Levesque's (1984) terminology. It is now feasible to develop knowledge representation servers on a par with floating-point arithmetic units and numeric libraries, as software (and perhaps ultimately hardware) modules with well-defined functionality and fast, reliable performance. As with arithmetic units, such servers by no means solve all the problems of a particular application domain, but they do greatly reduce the burden of system development, allowing effort to be focused on the specifics of particular systems rather than on what should be general utilities.

Figure 1 shows the architecture of KSSn as a family of modules around the knowledge representation server, KRS.

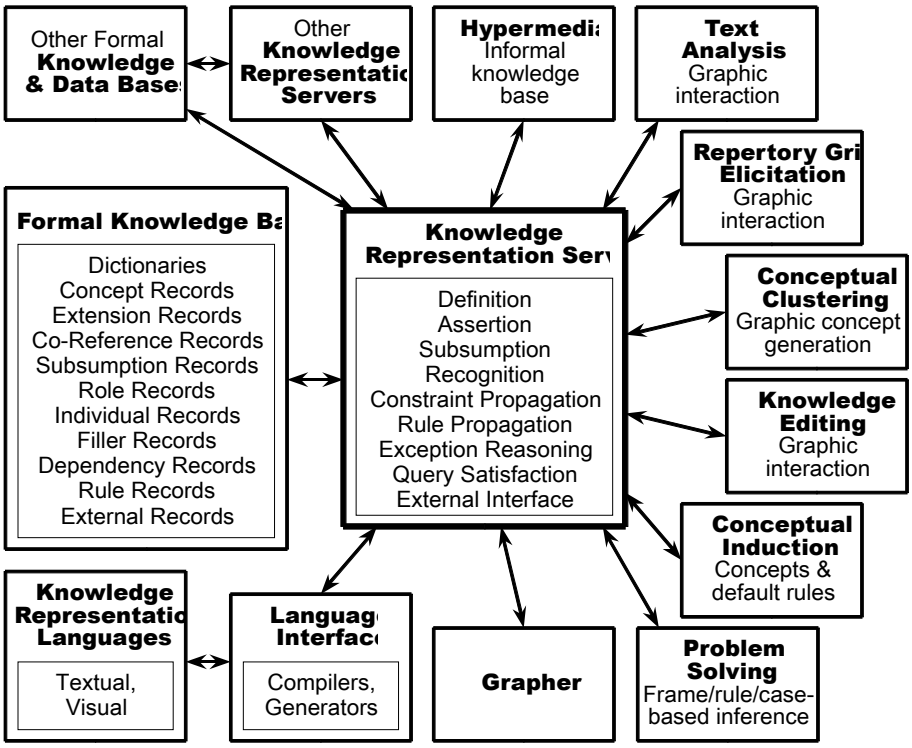

Figure 1 Architecture of the knowledge representation system

The modules are (reading clockwise from the top left):

- Interface modules to other knowledge bases and servers, including databases.
- A hypermedia module allowing informal knowledge structures in text and images to be captured, accessed and linked.
- A text analysis module allowing documents to be analyzed and associations between significant words to be graphed.
- A repertory grid expertise transfer module allowing graphic definition of concepts and graphic creation and editing of individuals.
- A conceptual clustering module allowing interactive definition of new concepts.
- A knowledge editing module allowing the interactive development and editing of knowledge structures through a visual language.
- A conceptual induction module creating rules about specified subsets of individuals and transforming them to a minimal set of concepts and default rules.
- A problem solving module supporting frame, rule and case-based inference from the knowledge structures.
- A grapher laying out specified concept structures and individual structures.
- A language interface accepting and generating definitions and assertions in formal knowledge representation languages, both textual and visual.

The knowledge representation services of KRS, the central server module, correspond to those of CLASSIC (Borgida, *et al*, 1989), augmented with inverse roles, data types for integers, reals, strings and dates, and with rule representation that allows one rule to be declared an exception to others. KRS may be seen as providing a fast and principled implementation of a frame/rule knowledge representation and inference engine capable of operating with large knowledge bases.

OBJECT-ORIENTED IMPLEMENTATION

The implementation of KSSn as a class library in an object-oriented language provides a simple software architecture based on a few, verifiably principled modules. Figure 2 shows the objects used in the server implementation.

The basic class library provides variable-length record objects which are sub-classed to provide dictionaries, sets and concept and individual records. The objects are:

- *Concept records* hold the constraints defining a concept: its primitives and those disjoint from them; its extensions as the subsets of individuals that must be, and may be, included in any subset recognized; its cardinality and co-reference constraints on any subset recognized; its rule concepts, if any, which will be asserted of any individual recognized; and, for each role constrained in the concept, the concept constraining it.
- *Individual records* hold the templet concept defining intensional constraints upon an individual and, for each role constrained in the templet or filled in the individual, a specification of the role fillers.
- *Filler records* hold subsets of individuals that fill slots.
- *Dictionaries* recognize concept, role and individual names and generate an accession number used to index other structures. There is a clear separation between concepts, roles and individuals and a lexical item can occur in more than one dictionary with different meanings.
- *Extension records* hold subsets of individuals, typically those that may occur in a role filling set and those that must occur in it.
- *Data records* hold subsets of external individuals that are explicitly recognized by the server, notably integers, reals, dates and strings.
- *Co-reference records* hold equivalence subsets of role chains expressing co-reference constraints.

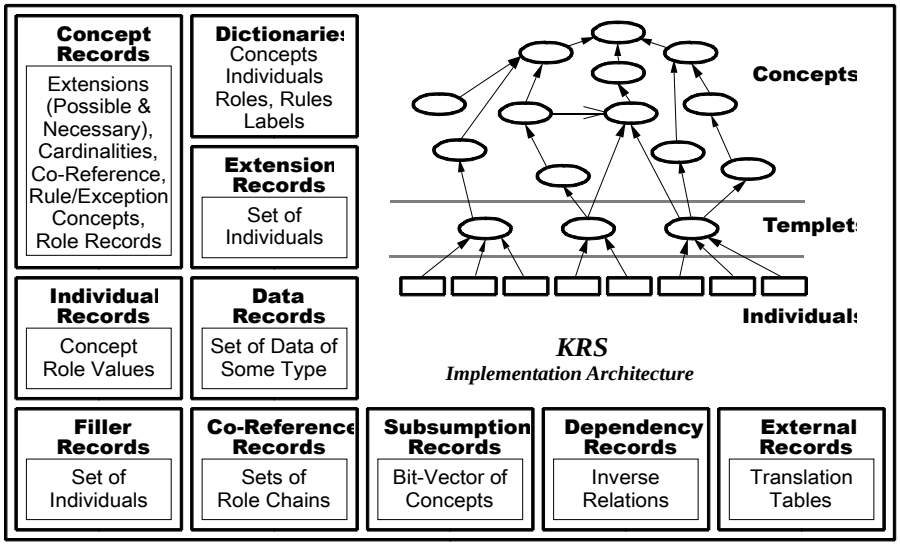

Fig.2 Data structures in the knowledge representation server

• *Subsumption records* hold computed subsumption relations on a three-valued basis, that one concept subsumes another, does not, or cannot.
• *Dependency records* keep track of inverse relations between individuals, supporting explicit inverse roles, efficient rule propagation and efficient retraction.
• *External records* hold tables translating between internal and external concepts and individuals.

KDRAW VISUAL LANGUAGE

An important component of KSSn in the context of this paper is the graphic knowledge editor, KDraw, at the right of Figure 1. This is a drawing tool designed for ease of use that provides a visual structure editor for semantic networks representing classes, objects and rules in KRS. Nosek and Roth (1990) have demonstrated empirically that the visual presentation of knowledge structures as semantic nets leads to more effective human understanding than does textual presentation of the same structures. KDraw is a formal visual language that corresponds exactly to the underlying algebraic semantics of KRS that has remarkably few visual primitives and is easily learnt and understood (Gaines, 1991c). The best way to introduce it here is through examples drawn from an orgainzational application. Figure 3 shows the top level conceptual structures the domain (Voß, Karbach, Drouven, Lorek & Schuckey, 1990).

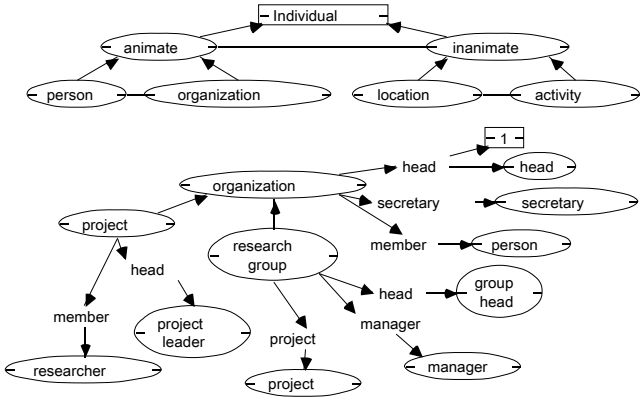

Fig.3 Top-level and organizational ontologies in KDraw

The visual language used is precisely defined. Concepts are ovals, primitive concepts are ovals with small horizontal lines inside each side, individuals rectangles, roles unboxed text, rules rounded-corner boxes, constraint expressions rounded-corner boxes with small horizontal lines. Lines without arrows connecting primitive concepts denote that the concepts are disjoint, and connecting roles that they are inverse. The interpretation of the arrows in the editor is overloaded but well-defined by the types of the objects at their head and tail, e.g.:

concept → concept	definitional subsumption
concept → role → concept	definitional role with conceptual constraint
concept → role → constraint	definitional role with extensional, cardinality or numeric constraint
constraint → individual	extensional constraint
individual → concept	asserted constraint on individual
individual → role → individual	asserted value of role for individual
concept → rule → concept	production rule
rule → rule	first rule exception to second

Thus the top knowledge structure in Figure 3 defines "animate" and "inanimate" to be disjoint primitive concepts of type "Individual"; "person" and "organization" to be disjoint concepts inheriting from "animate"; and "location" and "activity" to be disjoint concepts inheriting from "inanimate". The lower knowledge structure further defines "organization" (note that nodes may be freely duplicated for the sake of visual appearance) to have the role "head" filled by exactly one individual of type "head", to have the role "secretary" possibly filled by individuals of type "secretary", and to have the role "member" possibly filled by individuals of type "person". A project, since it is shown to inherit from "organization" also has these roles and constraints but it is shown to be further constrained to have its "head" role filled by a "group head" and its "member" role filled by individuals of type "researcher".

Figure 4 shows the complete knowledge, data and problem solving structures in the visual language as a screen dump from the KDraw editing window. A "fit-to-size" command was used to get the structures into a small diagram for this paper. The problem is one of room allocation derived from an ESPRIT project that has recently been placed in the public domain as part of Project Sisyphus. This is research program to encourage international collaboration in knowledge-based system development. A number of problem datasets have been made available through Sisyphus, and a major part of the EKAW'91 program was devoted to reports on the solution of these problems using different approaches and techniques (Linster, 1991).

Human-computer interaction in the editor is modeled on Apple's MacDraw with additional features appropriate to the language such as arcs remaining attached to nodes when they are dragged. A popup menu as shown at the bottom of Figure 4 allows connecting lines to be entered easily. The syntax of possible node interconnections and constraint expressions is enforced—it is not possible to enter a graph that is syntactically incorrect. Cut-and-paste of graphs and subgraphs is supported, and popup menus allow nodes to be connected with the minimum of effort. Updates are efficient and graphs with over a thousand nodes can be manipulated interactively. Scroll, zoom and fit-to-size facilities allow large data structures to be navigated easily. However, partitioning data structures over several screens is encouraged and has proved practical in managing large knowledge structures.

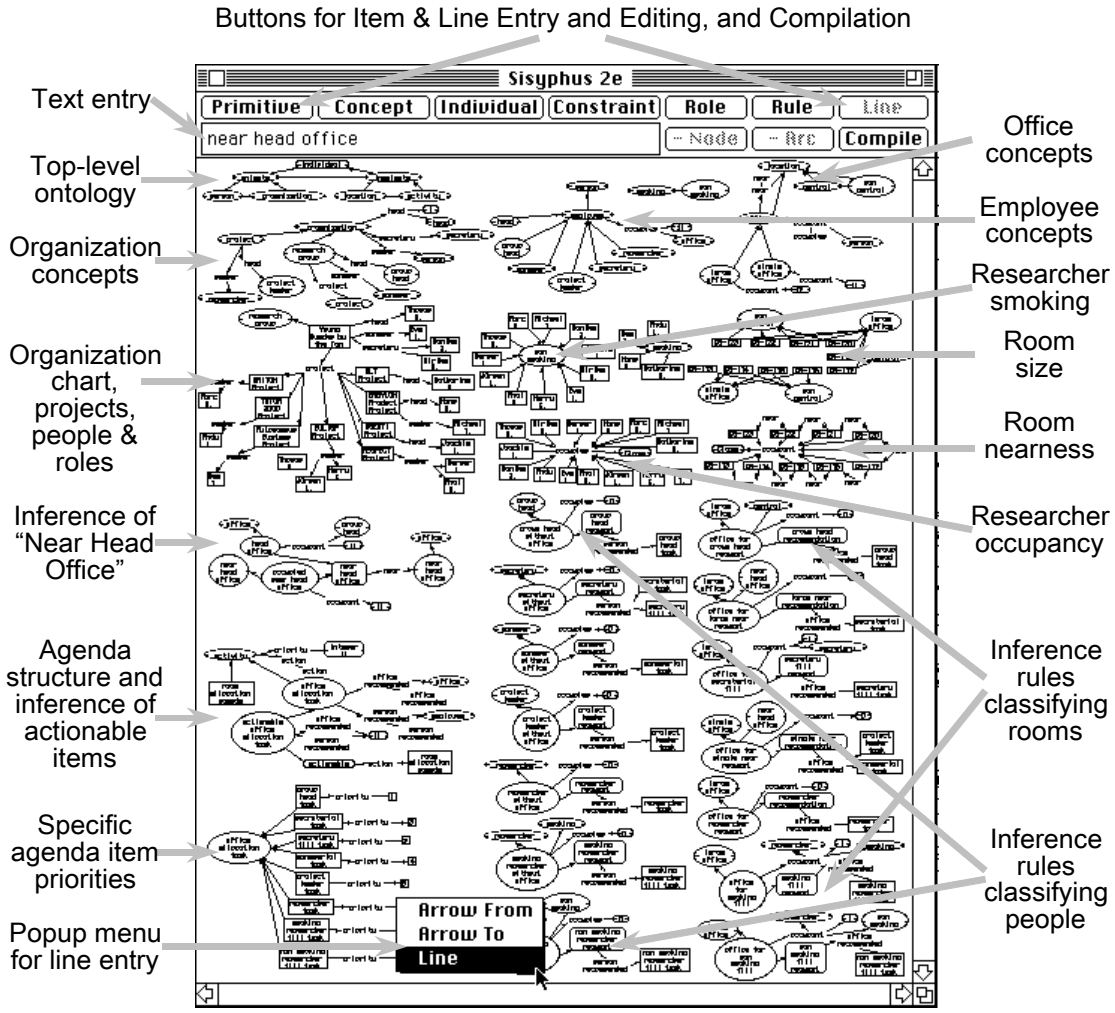

Figure 4 Organizational problem, its solution and example dataset expressed in KDraw visual language

CONCLUSIONS

It has been shown that open architecture, object-oriented knowledge representation servers are straightforward to implement using an object-oriented language, and offer an important technology for organizational integration in the development of computer-integrated manufacturing systems. The design philosophy is that an analogy to arithmetic units holds, and it is appropriate to develop clean, complete and fast units with limited capabilities, and then supplement them with other units providing additional functionality ranging from general theorem provers to highly domain-specific modules. The operation of the server with the functionality and optimizations described is fast and storage efficient, and was easily implemented. We have run it with knowledge bases of several thousand concepts and individuals from a variety of sources.

As already noted in the literature (De Troyer, Keustermans & Meersman, 1986), the object-oriented implementation contributes nothing directly to the object-oriented system being implemented. That is, there is no trace of the C++ classes in the KRS knowledge structures. What is contributed, as noted in the literature comparing C++ and Lisp (Trickey, 1988), is a superb software engineering environment supporting extremely well-structured and transparent implementation of conceptually difficult artificial intelligence knowledge representation and inference processes.

This difference shows up most clearly in issues such as multiple inheritance. We have found no need for multiple inheritance in the C++ implementation language. However, it is an essential feature of KRS. A simple explanation of this difference is that at the implementation level one has full conceptual control and it easy to incorporate the effects of multiple inheritance through appropriate structures in a single inheritance system. At the knowledge level, however, one is attempting to represent cognitive structures without distortion and multiple inheritance is common in human conceptual frameworks.

ACKNOWLEDGEMENTS

This work was funded in part by the Natural Sciences and Engineering Research Council of Canada. I am grateful to Marc Linster for the Sisyphus data set, and to Mildred Shaw for collaboration in research and comments on this paper.

REFERENCES

Borgida, A., Brachman, R.J., McGuiness, D.L. & Resnick, L.A. (1989). CLASSIC: a structural data model for objects. Clifford, J., Lindsay, B. & Maier, D., Eds. **Proceedings of 1989 ACM SIGMOD International Conference on the Management of Data**. pp.58-67. New York: ACM Press.

Brachman, R. J., Gilbert, V.P. & Levesque, H. J. (1985). An essential hybrid reasoning system: knowledge and symbol level accounts of KRYPTON. **Proceedings of IJCAI85**. pp.547-551. Los Altos: Morgan Kaufmann.

Brachman, R. J., & Levesque, H. J. (1984) The tractability of subsumption in frame-based description languages. **Proceedings of AAAI-84**. pp.34-37. San Mateo, California: Morgan Kaufmann.

Brachman, R.J. & Schmolze, J. (1985). An overview of the KL-ONE knowledge representation system. **Cognitive Science, 9**(2) 171-216 (April-June).

Davis, J.P. (1989). **The Knowledge Dictionary: A KBMS Architecture for the Many-to-Many Coupling of Knowledge Based-Systems to Databases**. PhD Thesis, University of South Carolina, UMI9005097

De Troyer, O., Keustermans, J. & Meersman, R. (1986) How helpful is an object-oriented language for an object-oriented database model? **Proceedings of 1986 International Workshop on Object-Oriented Database Systems**. pp.124-132. Washington: IEEE Computer Society Press.

Gaines, B.R. (1988a). Structure, development and applications of expert systems in integrated manufacturing. Kusiak, A., Ed. **Artificial Intelligence Implications for CIM**. pp.117-161. Bedford, UK: IFS Conferences.

Gaines, B.R. (1988b). Knowledge acquisition systems for rapid prototyping of expert systems. **INFOR, 26**(4), 256-285 (November).

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

Gaines, B.R. (1991b). Integrating rules in term subsumption knowledge representation servers. **AAAI'91: Proceedings of the Ninth National Conference on Artificial Intelligence**. pp.458-463. Menlo Park, California: AAAI Press.

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

Gaines, B.R. & Linster, M. (1990). Integrating a knowledge acquisition tool, an expert system shell and a hypermedia system. **International Journal of Expert Systems Research and Applications 3**(2) 105-129.

Gaines, B.R., Rappaport, A. & Shaw, M.L.G. (1992). Combining paradigms in knowledge engineering. **Data and Knowledge engineering.** to appear.

Gupta, R. (1991) **Object-Oriented Databases with Applications to CASE, Networks, and VLSI CAD.** Englewood Cliffs, New Jersey: Prentice-Hall.

Levesque, H. J. (1984) A logic of implicit and explicit belief. **Proceedings of AAAI-84**. pp.198-202. San Mateo, California: Morgan Kaufmann.

Linster, M., Ed. (1991) **Sisyphus Working Papers Part 2: Models of Problem Solving**. EKAW91, Glasgow: University of Strathclyde.

MacGregor, R.M. (1988) A deductive pattern matcher. **Proceedings of AAAI88**. pp.403-408. San Mateo: Morgan Kaufmann.

Nebel, B. (1990) **Reasoning and Revision in Hybrid Representation Systems**. Berlin: Springer-Verlag.

Nosek, J.T. & Roth, I. (1990) A comparison of formal knowledge representations as communication tools: predicate logic vs semantic network. International Journal of Man-Machine Studies 33, 227-239, 1990.

Sagawa, J.M. (1990). Repository manager technology. **IBM Journal of Research and Development 29**(2) 209-226.

Scheer, A.-W. (1989) **Enterprise-Wide Data Modeling.** Berlin: Springer.

Schmidt-Schauss, M. (1989) Subsumption in KL-ONE is undecidable **Proceedings of KR'89: First International Conference on Principles of Knowledge Representation and Reasoning.** pp.421-431. San Mateo, California: Morgan Kaufmann (May).

Scott, W.R. (1981) **Organizations: Rational, Natural and Open Systems.** Englewood Cliffs, New Jersey: Prentice-Hall.

Shaw, M.L.G. & Gaines, B.R. (1987). KITTEN: Knowledge Initiation & Transfer Tools for Experts & Novices. **International Journal of Man-Machine Studies, 27**, 251-280.

Shriver, B. & Wegner, P. (1987) **Research Directions in Object-Oriented Programming**. Massachusetts: MIT Press.

Trickey, H. (1988) C++ versus Lisp: a case study. **Usenix Procceding C++ Workshop**. pp.440-449. Berkeley: Usenix Association (November).

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