DOCUMENTS AS EXPERT SYSTEMS

Brian R Gaines & Mildred L G Shaw

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

Abstract: This paper is written in a document production tool that appears to a user as a word processor but also acts as an expert system shell with frame and rule representations supporting deductive inference. The electronic version of the document is active, providing typographic text and page layout facilities, versioning, hypermedia sound and movies, hypertext links, and knowledge structures represented in a visual language. It can be read as a hypermedia document and also interrogated as a knowledge-based system for problem-solving. The paper version of the document, which you are now reading, is produced by printing the electronic version. It loses its active functionality but continues to act as a record of the knowledge in the document. The overall technology has been developed as an alternative approach to the dissemination of knowledge bases. It also provides a different interface to knowledge-based systems that emulates document interfaces with which many users are already familiar.

1 INTRODUCTION

The knowledge document publication system emulates conventional word-processing packages as closely as possible to require the minimum of new skills in the user. It produces documents that are formatted and paginated for printing so that parallel publication of paper and electronic documents is available. It allows diagrams, pictures, video and sound to be integrated in documents, with their preparation and editing being based on existing packages so that again the user has the minimal learning requirements. For example, the picture inset in this paragraph is a QuickTime video with moving picture and sound commentary that may be played by double clicking in the picture. The mechanism for linking to multi-media material is also used to provide simple and versatile hypertextual linking. Formal knowledge structures may also be embedded in documents, represented as semantic networks in a visual language easily understood by people. The same knowledge structures may also be accessed through computer programs to provide the decision support and problem solving capabilities of an expert system.

Apollo 11 LiftOff

The knowledge document publication system is part of a research program on general 'knowledge support systems' integrating many information technologies to support knowledge processes in human society (Gaines, 1990). Since the intended users are not computer specialists, our objective has been to achieve all the functionality outlined above through an environment that appears simple and natural to the user. That is, the project is successful to the extent that the system appears like a conventional word processor rather than an over-engineered concatenation of multi-media and artificial intelligence functionality. This paper focuses on the expert system functionality of the system and illustrates this through some studies of organizational modeling and problem solving which were reported in greater detail in an (inactive) paper at the Conference on Organizational Computing Systems (Gaines, 1991d).

The problem considered is one of room allocation from an ESPRIT project (Voß, Karbach, Drouven, Lorek & Schuckey, 1990) that has recently been made part of Project Sisyphus. Sisyphus is a research program to encourage international collaboration in knowledge-based system development initiated by the European Knowledge Acquisition Workshop in 1989. 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).

What is remarkable about the document you are reading is that the paper version of it reports a solution to the problem, giving all the knowledge structures involved in visual form, and the electronic version of it *is* the solution. That is, if you opened this paper in the associated word processor you could interrogate it to solve room allocation problems. You could edit the visual knowledge structures within the document, for example by adding additional rules, and when you interrogated it again those rules would be in effect. Thus, the document itself provides an active, editable knowledge base and problem-solving inference engine.

The next section gives an overview of the architecture of the system, followed by the visual language for knowledge representation and the problem formulation and solution in this language.

2 KSSn ARCHITECTURE

KSSn (Knowledge Support System n) is the latest in a series of developments deriving from our initial implementation of KSS0 (Gaines, 1988a,b) 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, Rappaport & Shaw, 1989; Gaines & Linster, 1990). KSSn is designed as a C++ class library implementing KRS (Gaines, 1991a), a KL-ONE-like (Brachman & Schmolze, 1985) knowledge representation server, and a set of associated functional modules for knowledge elicitation, text analysis, empirical induction, graphic knowledge base editing, and so on.

Figure 1 shows the architecture of KSSn as a family of modules attached to the knowledge representation server, KRS. The modules are (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. The linkage structure is held as a knowledge base.
- A text analysis module allowing documents to be analyzed in terms of word usage, and associations between significant words to be graphed—based on TEXAN in KSS0. This enables protocols and technical documents to be used to initiate knowledge acquisition.
- A repertory grid expertise transfer module allowing graphic definition of concepts and graphic creation and editing of individuals—based on the elicitation screens of KSS0.
- A conceptual clustering module allowing interactive definition of new concepts—based on the hierarchical and spatial clustering from KSS0.
- 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—based on the Induct algorithm.
- A problem solving module supporting frame, rule and case-based inference from the knowledge structures.
- A grapher laying out specified parts of the concept subsumption graph, concept structures and individual structures—using an incremental layout algorithm that can be used interactively to support the production of clear visual knowledge 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. KWrite, the document production tool used to produce this paper, may be seen as providing a word processing user interface to the functionality shown in Figure 1.

3 KDRAW VISUAL LANGUAGE

An important component of KWrite and 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. We have developed 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 & Shaw, 1990).

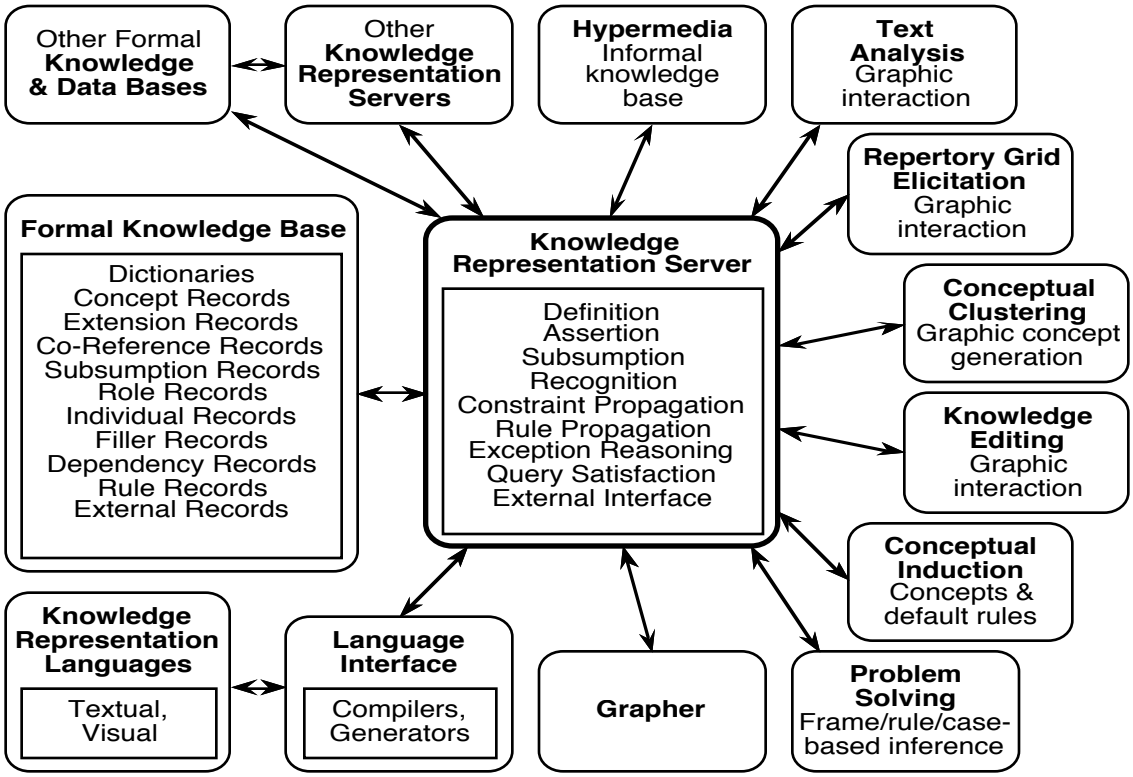


Figure 1 Architecture of the knowledge representation server

Visual representation of knowledge structures has been common since the early development of diagrams and taxonomies, and was associated with semantic networks in the early days of artificial intelligence (Quillian, 1968). The early development of such nets resulted in criticisms that the semantics of particular diagrams was not well-defined (Woods, 1975; Brachman, 1977). Nodes, arcs and their labels could be used very freely and ambiguously and diagrams were subject to differing interpretations. In the 1970s there were proposals for network formalisms with well-defined semantics (Cercone & Schubert, 1975; Fahlman, 1979; Brachman, 1979). However, these preceded two important developments in computing: first, the ubiquity of personal workstations with high resolution graphics supporting visual languages as operational editors (Glinert, 1990); second, the studies of complexity issues in knowledge representation, leading to the simplified and tractable semantics of CLASSIC (Borgida et al, 1989).

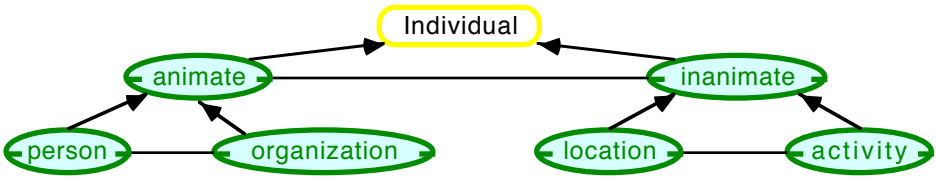
Computer production of visual forms of knowledge represented in a computer has been a topic of research since the early days of knowledge representation research (Schmolze, 1983) and a feature of many research systems (Kindermann & Quantz, 1989) and commercial products. Abrett and Burstein's (1988) KREME system graphically displays the computed subsumption relations between concepts so that those entering knowledge structures can see the consequences of definitions and detect errors due to incorrect or inadequate definitions.

The KDraw design (Gaines, 1991c) has drawn upon this previous research and experience to develop the visual syntax and underlying semantics of a visual language for term subsumption knowledge representation languages in the KL-ONE family. It focuses on the use of the language to enter and edit knowledge visually, and on its application in a highly interactive graphic structure editor. KDraw may be used just to support the entry of conceptual structures and facts in a knowledge base. However, it is capable of going beyond this and accepting problem solving knowledge structures in the form of concepts defining the premise and conclusion of rules. The rule structure provided in KDraw and KRS is itself powerful in both its representational and inference capabilities in supporting exceptions and defaults (Gaines, 1991b).

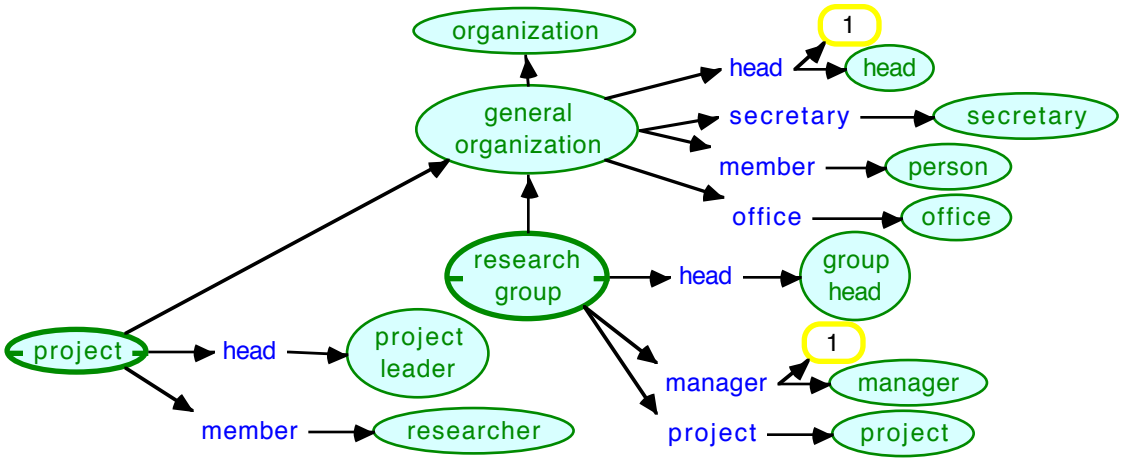
The structures below show the top level conceptual structures of an organizational domain. The visual language used in these diagrams is precisely defined. Concepts are ovals, primitive concepts are ovals with small horizontal lines inside each side, individuals are rectangles, roles are unboxed text, rules are rectangles with double lines at the sides, constraint expressions are rounded-corner boxes. Lines without arrows connecting primitive concepts denote that the concepts are disjoint, and those connecting roles denote that they are inverse. The interpretation of the arrows in the editor is overloaded but well-defined by the types of the objects at their head and tail, 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

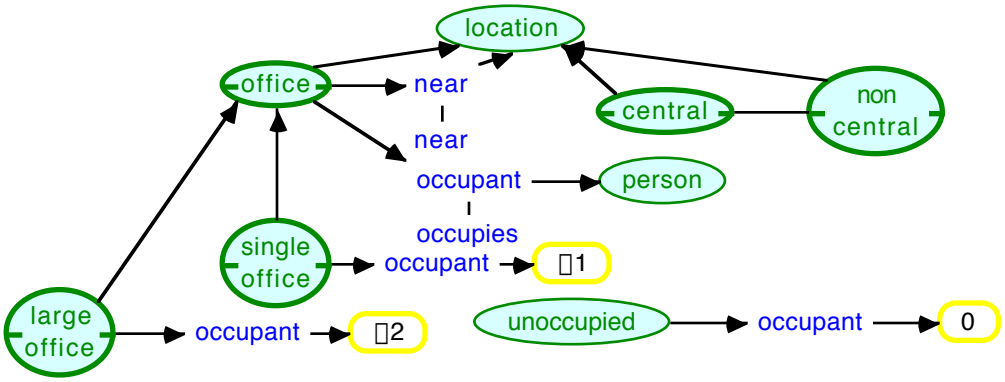
Thus knowledge structure 1 defines “animate” and “inanimate” to be disjoint primitive concepts of type “Individual”; “person” and “organization” to be disjoint concepts inheriting from “animate”; and “location” and “activity” to be disjoint concepts inheriting from “inanimate”. Knowledge structure 2 further defines “general organization” as an “organization” (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 is further constrained to have its “head” role filled by a “group head” and its “member” role filled by individuals of type “researcher”.



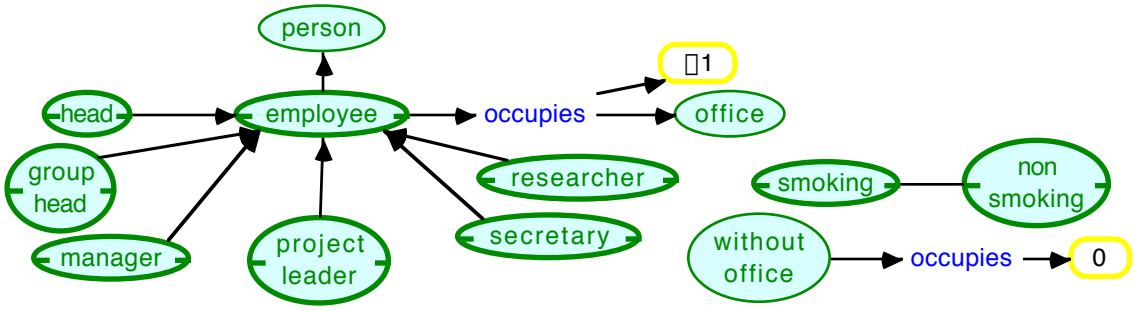
Structure 1: Top Ontology



Structure 2: Organization Ontology



Structure 3: Office Ontology



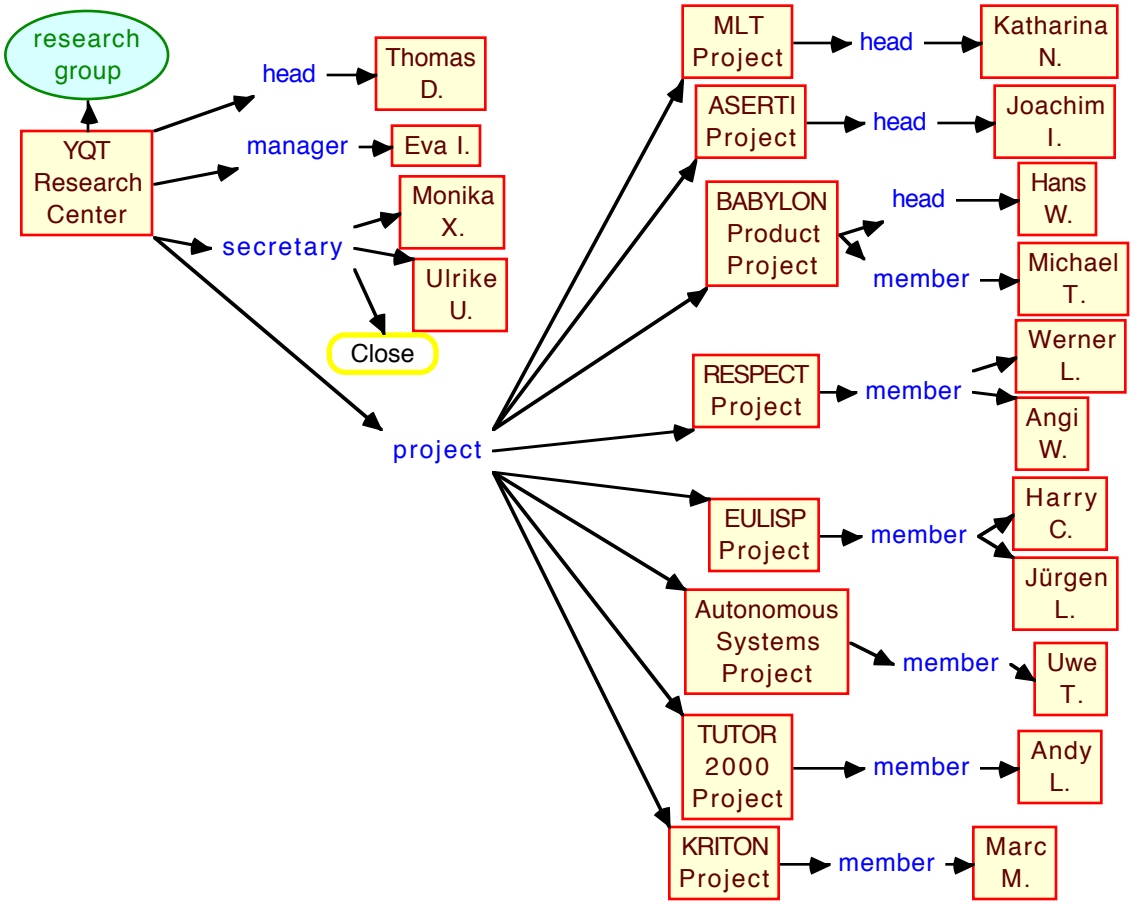
Structure 4: Employee Ontology

The underlying knowledge representation inference engine propagates constraints so that defining an individual as a "project" and then filling its "member" role with "Marc M." will automatically lead to "Marc M." being inferred to be a "researcher". Structure 4 shows some other features of the language. For example "occupant" and "occupies" are defined to be inverses, as are "near" and "near". Hence, if "Marc M." is asserted to fill the "occupant" role of office "C5-120" then it will be inferred that "C5-120" fills the "occupies" role of "Marc M.", and if "C5-119" is asserted to fill the "near" role of office "C5-120" then it will be inferred that "C5-120" fills the "near" role of "C5-119."

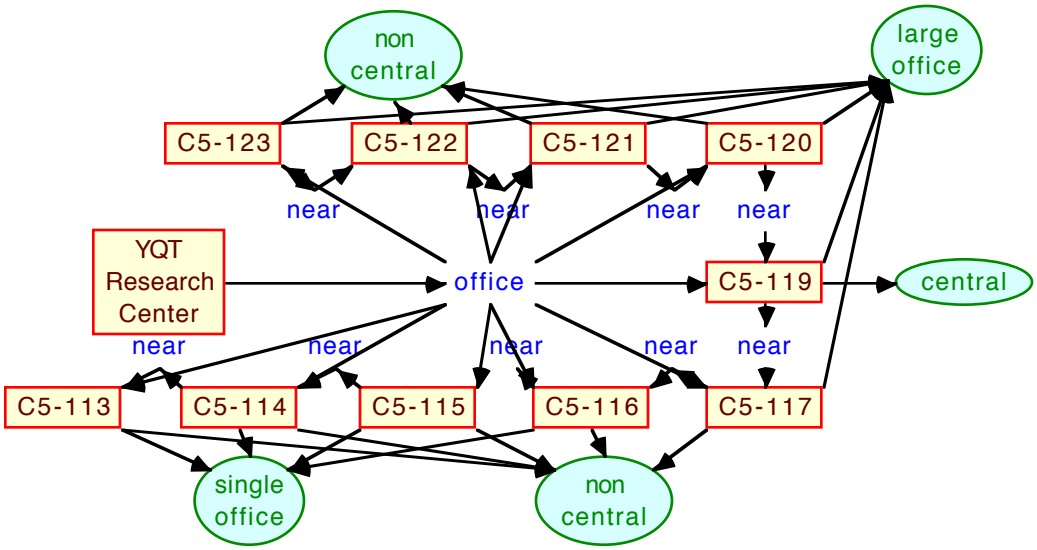
Structures 5 and 6 show the way in which facts may be asserted about individuals in the visual language. For example, the "RESPECT Project" in structure 5 is shown to be one of the fillers of the "project" role of "YQT Research Center" which is defined to be an instance of a "research group" defined in structure 3. This concept instantiation has the consequence that the conceptual constraints defined in structures 1 through 3 will be inherited appropriately by individuals in structure 5. For example, from structure 3 it will be inferred that "Thomas D." is "group head", "head", and the only filler of the "head" role, from structure 2 that he is an "employee" and a "person", and from structure 1 that he is "animate" and of type "Individual". Structure 6 supplies further facts about the project and the rooms that will be needed in the problem-solving activity.

These structures have a number of significant features as semantic networks. First, they are fully operational. They were created in the KDraw graphic structure editor and compile directly into knowledge structures in KRS. In this document they are active as well as operational, and can be edited with immediate impact on inference. Second, the visual language used is completely formally defined and intertranslatable with the underlying KL-ONE knowledge structures. Third, the freedom in layout has been used to create knowledge structures that are natural to the people involved. Structure 5 looks like an organization chart. Structure 6 is based on the actual room layout in the building. Fourth, editing these structures changes the ontologies and facts, and hence any related problem solving activity. For example, at the top right of the room layout, room "C5-119" is the only room asserted to be "central". If this appears to be restrictive when the room allocation rules are run then it is easy for the user to add arrows from "C5-117" and "C5-120" to"central" and see what changes result in the room allocation behavior.

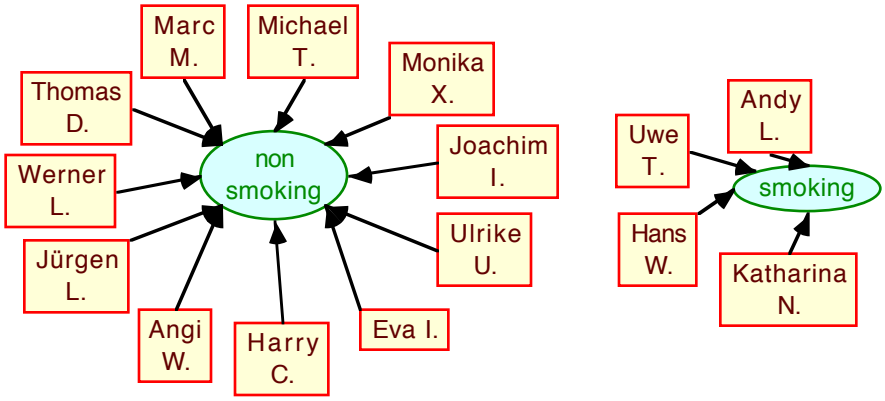
Double-clicking in the structures above brings up a floating dialog box allowing them to be edited as shown in Figure 2. Human-computer interaction in the editor is modeled on Apple's MacDraw with additional features appropriate to the language such as arcs remaining attached to nodes when they are dragged. A popup menu that appears when one mouses down on the right edge of a node 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 scroll and fit-to-size capabilities allow large structures with a thousand or more nodes to be edited in KDraw.



Structure 5: Organization Chart



Structure 6: Room Layout



Structure 7: Smoking Employees

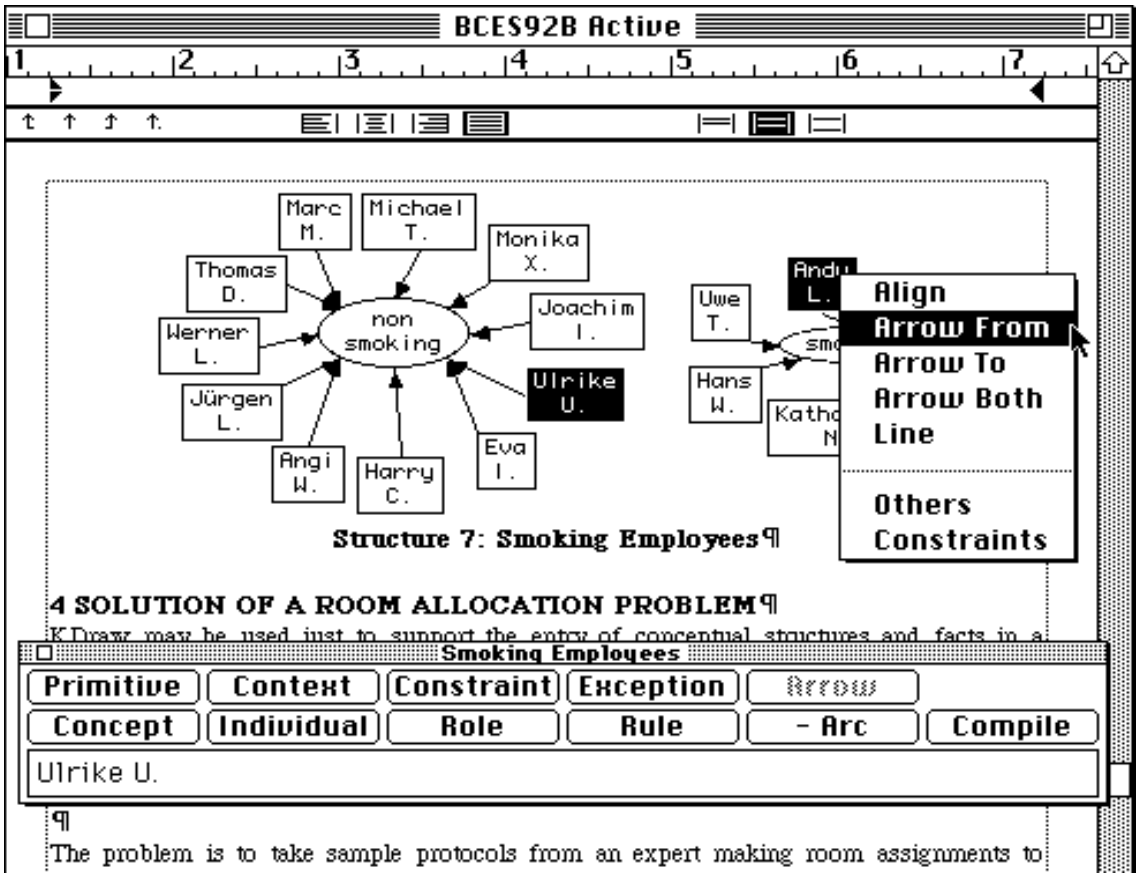


Figure 2 Screen dump of this paper being edited in KWrite

Double clicking on knowledge structure 7 above has brought up a floating dialog box to edit it. A popup menu has been accessed at the right edge of a particular graphic item in order to enter a connecting arrow. Some of the word-processing features of KWrite are also apparent in this figure.

4 SOLUTION OF A ROOM ALLOCATION PROBLEM

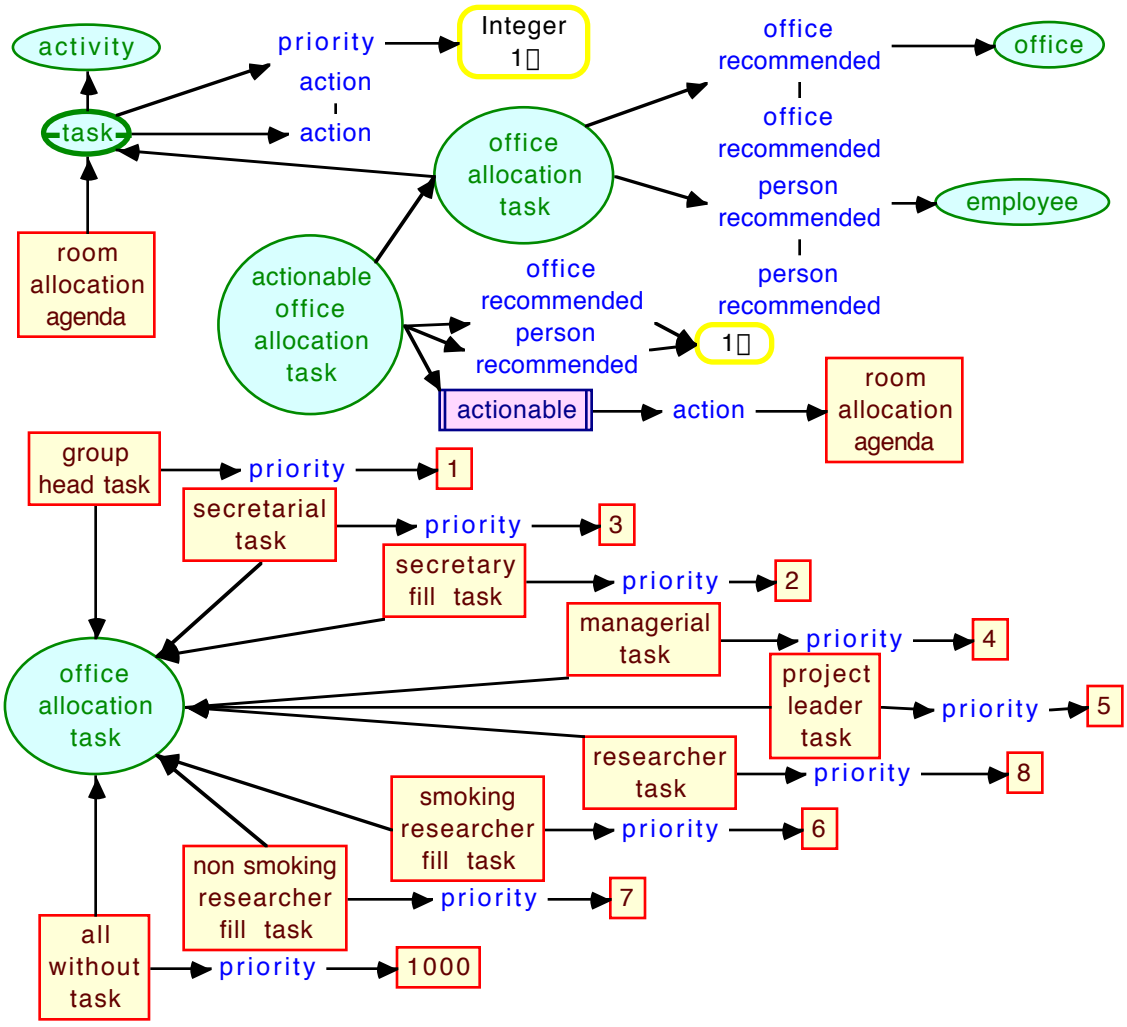
The problem is to take sample protocols from an expert making room assignments to researchers occupying a new building. The rules derived from the expert protocols are, in order of declining priority:

A large, central office should be allocated to the Head of Group.
A large office already occupied by a secretary should be allocated to another secretary.
A large office near the Head of Group is suitable for the secretaries.
A single office near the Head of Group is suitable for the Manager.
A single office near the Head of Group is suitable for a Project Leader.
An office with one smoking researcher occupant should be allocated to another smoking researcher.
An office with a non-smoking researcher occupant should be allocated to another non-smoking researcher.
A large office should be allocated to researchers.

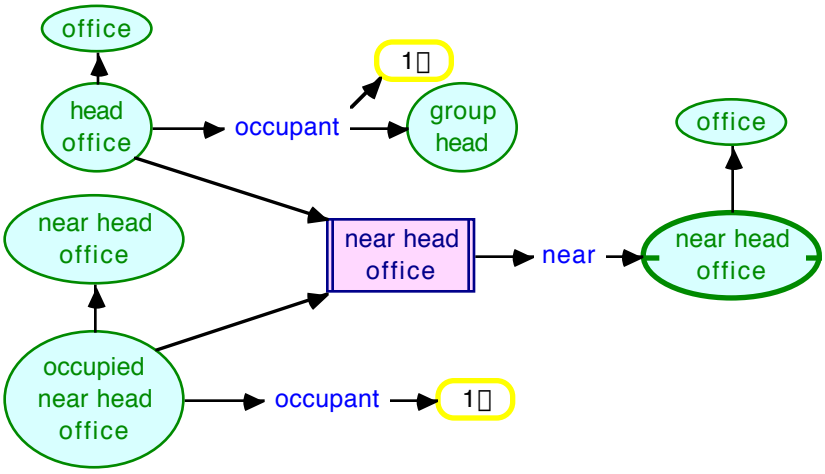
It is not sufficient just to implement these rules. Resource allocation problems tend to be either over-determined, and hence notionally insoluble, or under-determined, and hence subject to combinatorial explosion. People deal with this by problem reformulation, which is a strongly knowledge-based process, and this is expedited by systems that generate meaningful partial solutions that indicate the sources of obstacles to solution. What is required is a system that uses the rules to suggest allocations, allowing the user to make choices when the problem is under-determined, and to resolve conflicts when the problem is over-determined. The system should support retraction and backtracking if the user wishes to explore alternative solutions, perhaps involving considerations not expressed in the knowledge base. An agenda mechanism to support this approach is itself programmed as knowledge structure 8, with the concept of a prioritized task defined at the top and appropriate tasks and priorities defined at the bottom.

Structure 9 shows a rule to determine if a room is “near” to that of the “group head”. The upper left concept defines a “head office” as an “office” with at least one occupant and every occupant a “group head”. The rule “near head office” asserts that an office “near” such an office is “near head office”—the concept “near head office” will be asserted of it. The lower left concept defines an “occupied near head office” office as one which is “near head office” and has at least one occupant. Such an office again has its neighbors classified as “near head office”. This captures the expert’s reclassification of offices further away from head office as being “near” to it as intervening offices are filled.

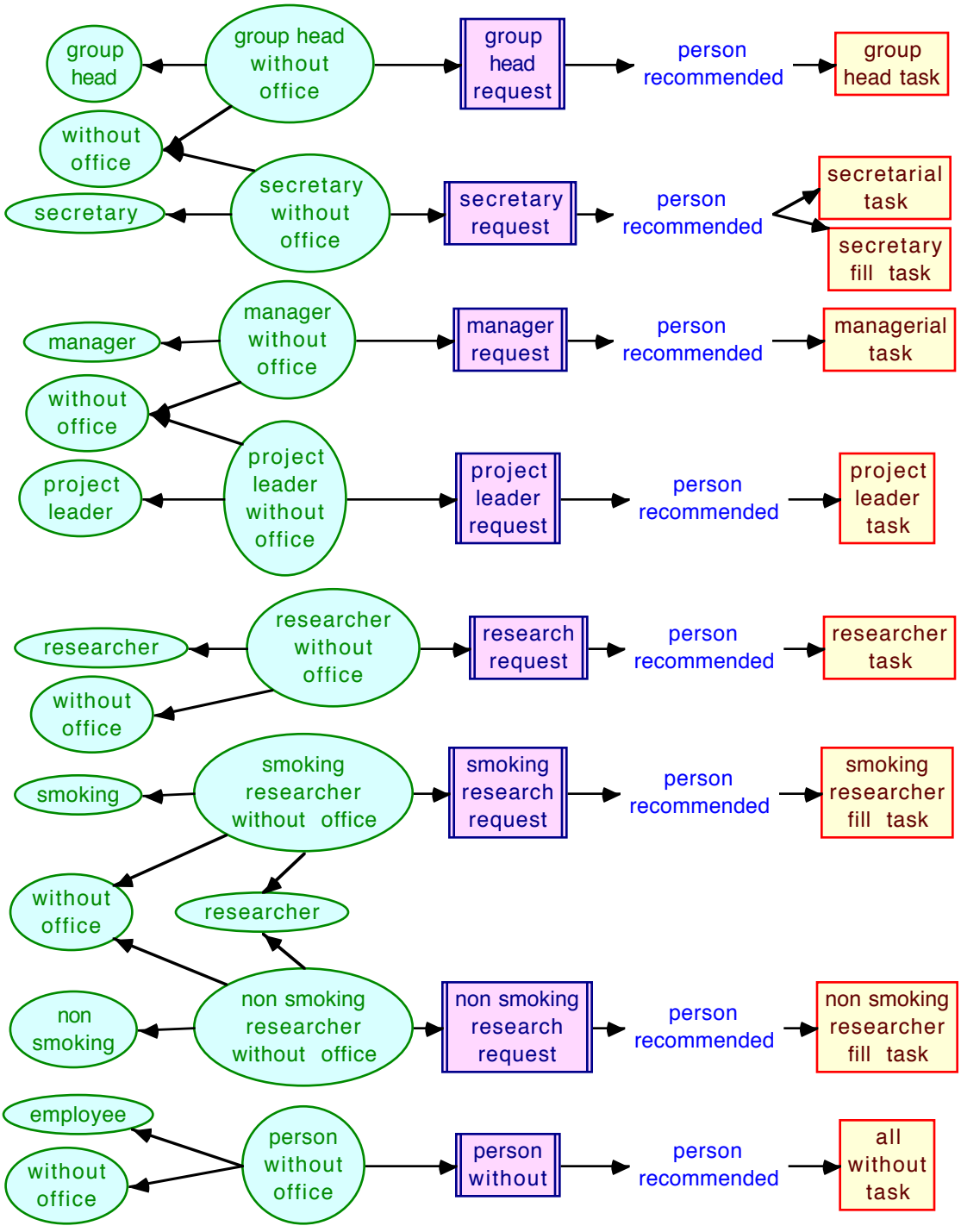
Structures 10 and 11 show the representation of the rules above. For example, at the top of structure 10, if an individual is a “group head” and “without office” (both defined in structure 2) then that individual is classified as “group head without office” and the “group head request” rule fires placing “group head task” in the “person recommended” role of the individual. Hence, since this role is defined as self-inverse in structure 8, the individual is inferred to be in the “person recommended” role of “group head task”.



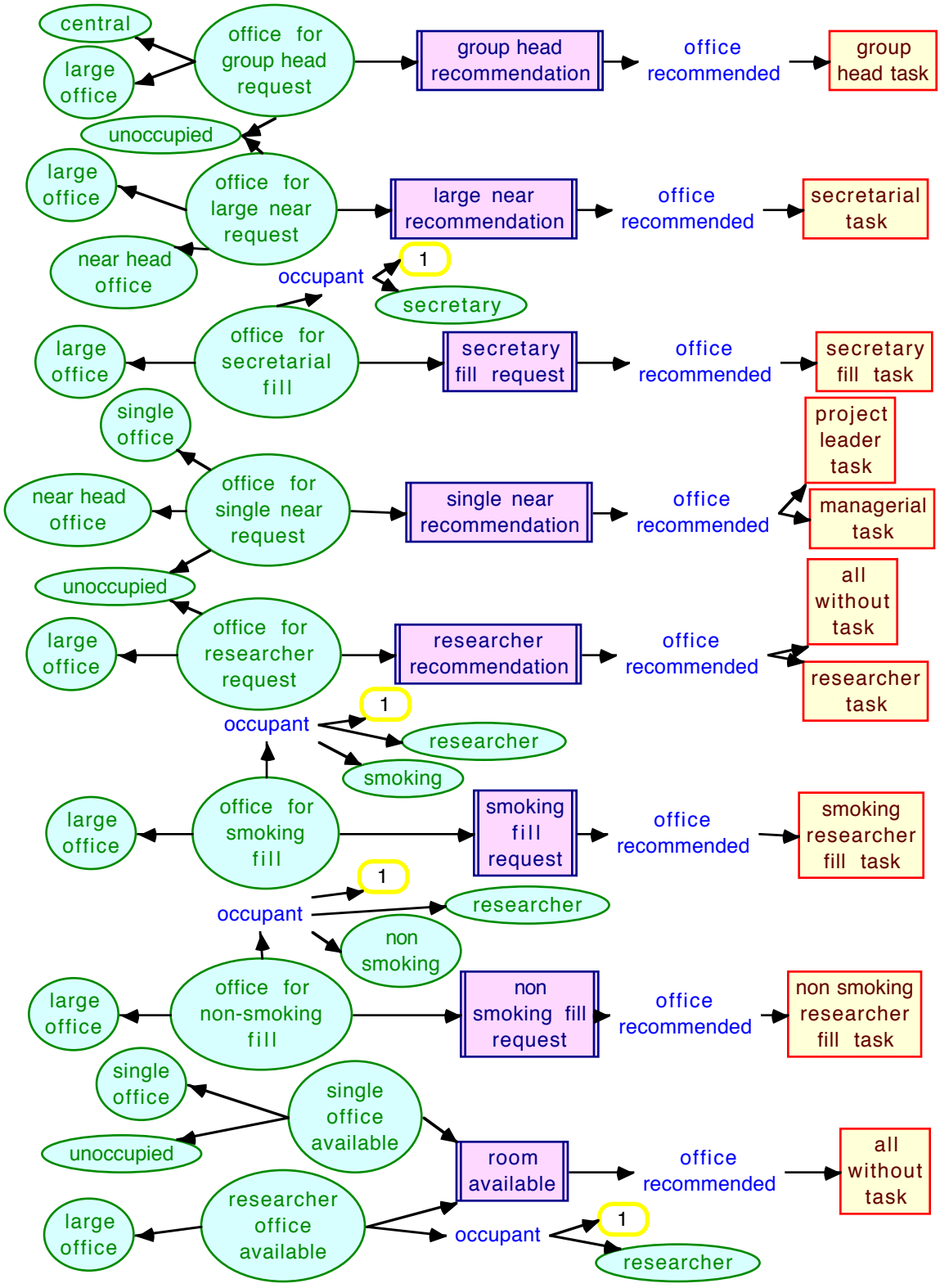
Structure 8: Agenda Mechanism



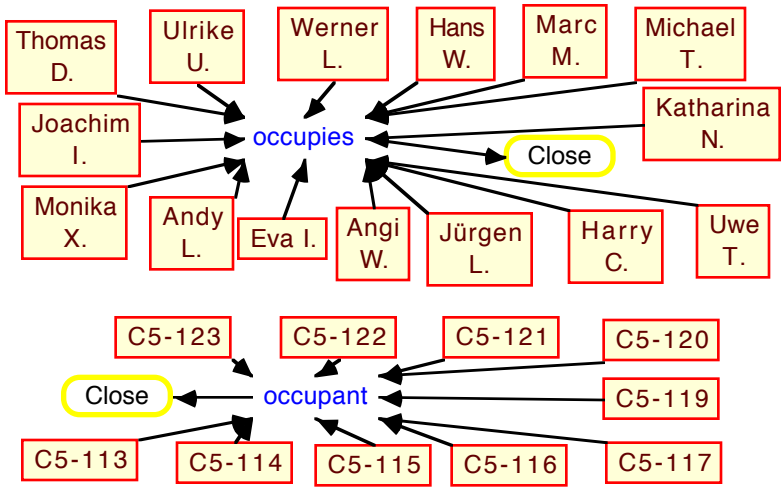
Structure 9: Rule for Room "Near Head Office"



Structure 10: Rules for Employee‘s Office Requirements



Structure 11: Rules for Office Suitability



Structure 12: Initial State of Employees and Offices

Structure 12 shows an initial state for problem solving when no one has been allocated a room and no rooms are occupied (the roles have been closed with no fillers). Other initial states are possible corresponding to partial allocation. All of the knowledge structures required for the room allocation problem are in this paper. They provide the user interface for domain, problem-solving and particular problem description. It remains to describe the user interface for problem solving. KRS is a server providing problem-solving capabilities, and KWrite and KDraw provide knowledge entry and editing sub-systems for creating a knowledge base. It is possible to use either textual or graphic querying of KRS to solve a particular problem. However, for the Sisyphus example the use of HyperCard to provide an open architecture user interface to KRS was demonstrated. Figure 3 shows the overall architecture.

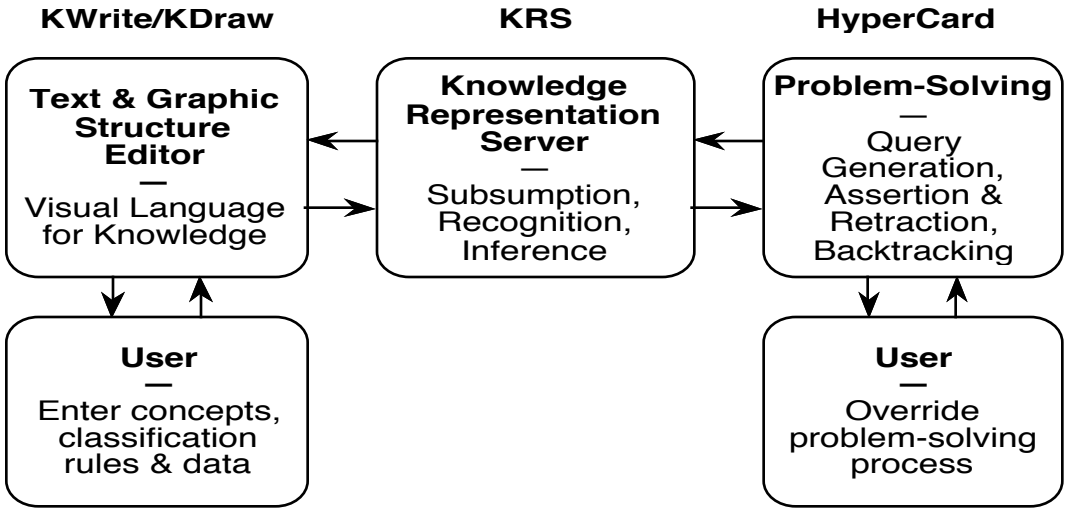


Figure 3 System architecture, functions and user interaction

HyperCard communicates with KRS, as do KDraw and KWrite, through Apple's System 7 inter-application communication protocol. This allows a server instantiation to run anywhere on a network and be accessed locally or remotely. The electronic version of this document serves as the knowledge base for KRS, and the visual knowledge structures shown are compiled directly into concepts, roles, individuals and rules in KRS.

Figure 4 shows the initial screen in HyperCard. The fields at the lower left list the knowledge bases embedded in this paper that will be used in problem solving. The user clicks on the "Solve" button to load these into a KRS server and commence inference. A sequence of recommended room allocations is then shown. If the problem is under-determined, this will involve some choice. Figure 5 shows the screen when the group head has been allocated and now the secretarial allocations are possible and top priority.

The person and room recommended for allocation are highlighted but users can over ride these by clicking on their own choices if they wish. Clicking on the "Allocate" button sends a message to KRS allocating the highlighted room to the highlighted person. The windows on the right in Figure 5 show all the employees without rooms and all the rooms available so that the user can override the recommendation process completely if desired. The user can also go to the screen shown in Figure 6 at any time, see the allocations already made, select any number of them and retract them. The KRS truth maintenance system automatically undoes any conclusions based on retracted data.

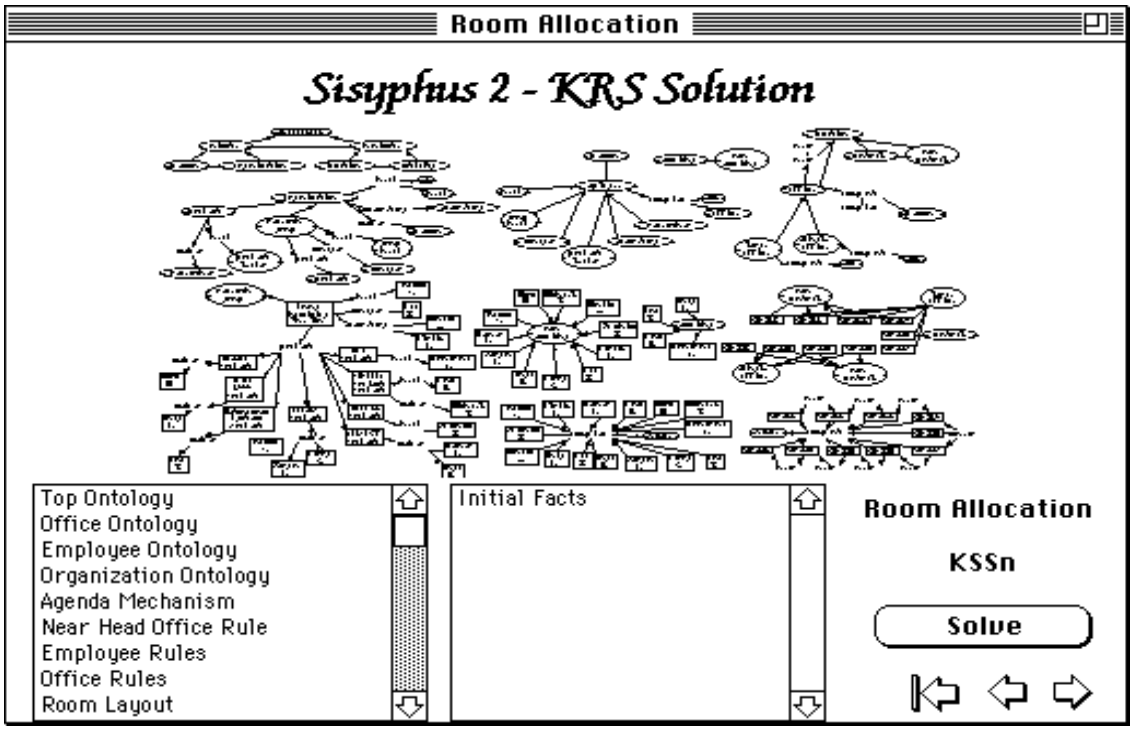


Figure 4 Initial HyperCard screen

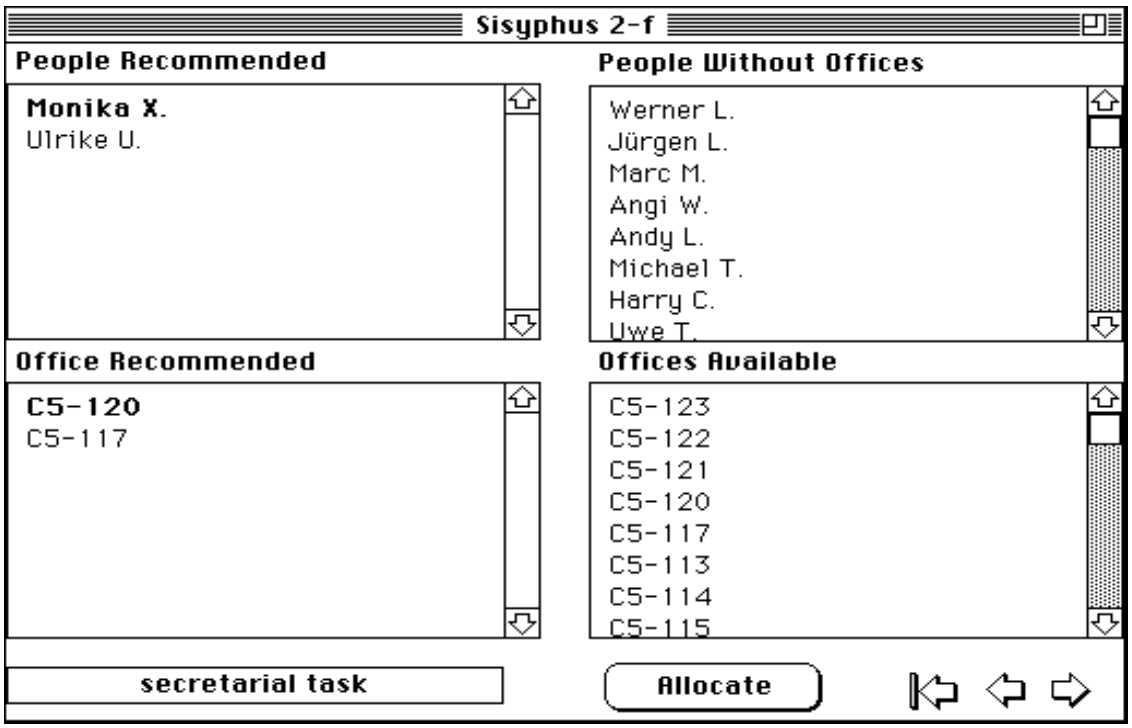


Figure 5 Secretarial room allocation recommendations

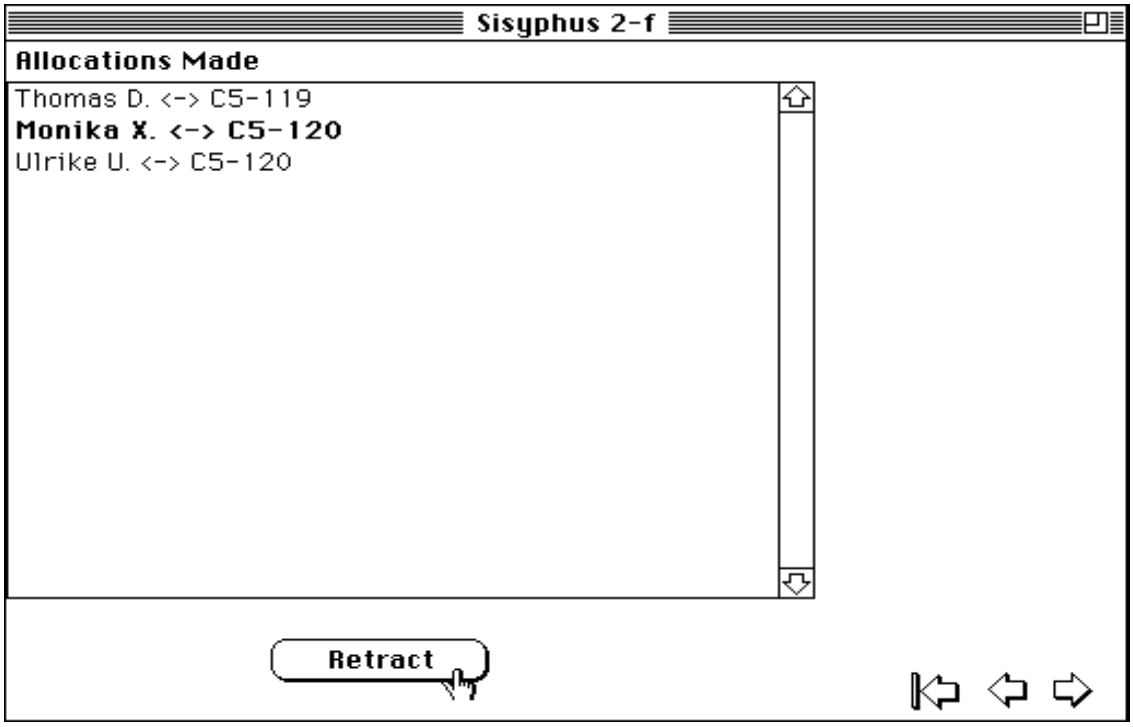


Figure 6 Allocations and retraction

4 CONCLUSIONS

This paper is written in a document production tool that appears to a user as a word processor but also acts as an expert system shell with frame and rule representations supporting deductive inference. The electronic version of the document is active, providing typographic text and page layout facilities, versioning, hypermedia sound and movies, hypertext links, and knowledge structures represented in a visual language. It can be read as a hypermedia document and also interrogated as a knowledge-based system for problem-solving. The paper version of the document, which you have just read, is produced by printing the electronic version. It loses its active functionality but continues to act as a record of the knowledge in the document. The overall technology has been developed as an alternative approach to the dissemination of knowledge bases. It also provides a different interface to knowledge-based systems that emulates document interfaces with which many users are already familiar.

In addition to demonstrating the knowledge document system, this paper has reported some experience in the design and implementation of a lightweight, object-oriented knowledge representation server, and its application to organizational modeling and problem solving. The primary user interface is through a formal visual language implemented simply and naturally as a drawing environment on graphic workstations. The open architecture implementation of the server allows it to be integrated with existing applications, such as corporate database and accounting systems, and also allows additional functionality to be added through self-contained modules requiring no changes in the kernel system.

The problem solving example given has shown how knowledge entered visually can be used to model organizational structures in a way that is simple and natural, and leads directly to operational problem solving. The process described here has the following features:

- The knowledge and data structures are totally overt and easily edited
- The knowledge document format allows a single instance of them to be disseminated both as an active knowledge base and as a passive paper report
- The visual language allows knowledge associated with structures such as the organization chart and room layout to be presented very naturally
- The problem solving strategy is incremental and can be applied to extend an existing partial solution
- The arbitrary choices that arise in undetermined problems can be made by the system or made by a person with, perhaps, additional considerations in mind
- Condensed and understandable information is available through the agenda items to support an attractive presentation of the problem solving process to the user
- The solution developed is a highly generic problem solving strategy
- The various components of the solution may be envisioned as coming from different archives in a corporate knowledge repository

ACKNOWLEDGEMENTS

Financial assistance for this work has been made available by the Natural Sciences and Engineering Research Council of Canada. We are grateful to many colleagues for discussions that have influenced the research described. We are particularly grateful to Marc Linster, the GMD and the Esprit Reflect Project for making the dataset available. The research reported here would not have been possible without access to the word processing and page makeup software developed by Gary Crandall of Datapak, and much of the significant document processing functionality is a consequence of his work.

REFERENCES

Abrett, G. & Burstein, M.H. (1988) The KREME knowledge editing environment. In Boose, J.H. & Gaines, B.R., Eds. Knowledge Acquisition Tools for Expert Systems. pp.1-24. London, Academic Press.

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. (1977) What's in a concept: structural fondations for semantic nets. International Journal of Man-Machine Studies 9, 127-152.

Brachman, R.J. (1979). On the epistemological status of semantic nets. In Findler, N.V., Ed. Associative Networks: Representation and Use of Knowledge by Computers. pp.3-50. New York: Academic Press.

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

Cercone, N. & Schubert, L. (1975). Towards a state-based conceptual representation. Proceedings of AAAI75. pp.83-90. Los Altos: Morgan Kaufmann.

Crandall, G. (1990). Word Solution Engine Programmer's Manual. Vancouver, Washington, DataPak.

Fahlman, S.E. (1979). NETL: A System for Representing and Using Real-World Knowledge. Cambridge, Massachusetts: MIT 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. (1990). Knowledge support systems. Knowledge-Based Systems 3(3) 192-203.

Gaines, B.R. (1991a) Empirical investigation of knowledge representation servers: design issues and applications experience with KRS. AAAI Spring Symposium: Implemented Knowledge Representation and Reasoning Systems. pp. 87-101. Stanford (March)—SIGART Bulletin 2(3), 45-56 (June).

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/MIT Press (July).
Gaines, B.R. (1991c). An interactive visual language for term subsumption visual languages. IJCAI'91: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence. San Mateo, California: Morgan Kaufmann.
Gaines, B.R. (1991). Organizational modeling and problem solving using an object-oriented knowledge representation server and visual language. COCS'91: Proc. of Conference on Organizational Computing Systems. pp.80-94. ACM Press.
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. (1989). A heterogeneous knowledge support system. Boose, J.H. & Gaines, B.R., Eds. Proceedings of the Fourth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop. pp.13-1-13-20. Banff (October).
Gaines, B.R. & Shaw, M.L.G. (1990) Cognitive and logical foundations of knowledge acquisition. Boose, J.H. & Gaines, B.R. (Eds) Proceedings of the Fifth AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop. pp. 9-1-9-25. Banff (November).
Glinert, I.P., Ed. (1990) Visual Programming Environments: Paradigms and Systems. Los Alamitos, California: IEEE Computer Society Press.
Kindermann, C. & Quantz, J. (1989) Graphics-oriented user interfaces for KL-ONE. KIT Internal Report 23. Technical University of Berlin.
Linster, M., Ed. (1991) Sisyphus Working Papers Part 2: Models of Problem Solving. EKAW91, Glasgow: University of Strathclyde.
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.
Quillian, M.R. (1968). Semantic memory. Minsky, M, Ed. Semantic Information Processing. pp.216-270. Cambridge, Massachusetts: MIT Press.
Schmolze, J. (1983). KLONEDRAW—a facility for automatically drawing pictures of KL-ONE networks. Research in Knowledge Representation for Natural Language Understanding. pp.41-44. Report No.5421, Cambridge, Massachusetts: Bolt Beranek and Newman Inc.
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.
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).
Watanabe, H. (1989). Heuristic graph displayer for G-BASE. International Journal of Man-Machine Studies, 30(3) 287-302 (March).
Woods, W.A. (1975) What's in a link: Foundations for semantic networks. Bobrow, D.G. & Collins, A.M. (Eds) Representation and Understanding: Studies in Cognitive Science. pp.35-82. New York: Academic Press.