SHORT DESCRIPTION OF THE RESEARCH GROUP on

Knowledge-Based Systems (KBS)
and
Human-Computer Communication (HCC)

Gerhard Fischer, Andreas Lemke, Hal Eden, Anders Morch,
Andreas Girgensohn, Scott Henninger, Tom Mastaglio,
David Redmiles, Brent Reeves, Curt Stevens,
Kumiyo Nakakoji, Jonathan Ostwald,
Akihiro Shinmmori, and Frank Shipman

CU-CS-413-88 November 1989

Department of Computer Science
Campus Box 430
University of Colorado,
Boulder, Colorado, 80309

This research is supported by grants and contributions from the following organizations: National Science Foundation, Army Research Institute, Colorado Institute for Artificial Intelligence, USWest Advanced Technologies, NYNEX, Software Research Associates, AT&T, MCC (Austin, TX), NCR (Ft.Collins, CO),and Symbolics.

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION, UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

University of Colorado at Boulder

Department of Computer Science

ECOT 7-7 Engineering Center
Campus Box 430
Boulder, Colorado 80309-0430
(303) 492-1592

November 1989

Short Description of the Research Group on

Knowledge-Based Systems (KBS) and Human-Computer Communication (HCC)

Members

Gerhard Fischer (Professor); **Andreas Lemke** (Post Doctoral Fellow); **Hal Eden** (Professional Research Assistant); **Andreas Girgensohn, Scott Henninger, Tom Mastaglio, David Redmiles, Brent Reeves,** and **Curt Stevens** (Graduate Research Assistants); **Kumiyo Nakakoji, Jonathan Ostwald, Akihiro Shinmori**, and **Frank Shipman** (PhD and Master Students).

Global Research Goals

Our goal is to establish, both by theoretical work and by building prototype systems, the scientific foundations for the construction of intelligent systems which serve as amplifiers of human capabilities (e.g., to expand human memory, augment human reasoning, and facilitate human communication). A prerequisite for intelligent systems is that we understand the information processing possibilities and limitations of the human *and* the computer. We apply basic, qualitative theories of human thinking to guide the design of innovative systems. Our systems should not only be significant as technical achievements in computer science, but also because they are based upon principled analyses of how one can best help people to cope with complex information systems. Working in intelligent systems, it is not sufficient to know how to build these systems; one must discover which systems are worth building.

Knowledge-Based Systems (KBS) and Human-Computer Communication (HCC) are two crucial research areas for these goals. We are especially interested in understanding the possibilities of pursuing these two research areas together. The rationale for this approach is that on the one hand effective human-computer communication is more than creating attractive displays on a CRT screen: It requires providing the computer with a considerable body of relevant world knowledge as well as knowledge about the psychological characteristics and understanding of the user; on the other hand, the use of knowledge-based systems and expert systems will be severely limited if we are unable to eliminate the communication bottleneck.

In our current research effort, innovative system development, cognitive theory construction, and evaluation are combined and centered around the following themes: cooperative problem solving, user-centered design, intelligent support systems (e.g., critics, advisors), intelligent information management, construction kits and design environments, human problem-domain communication, explanation and visualization components, and reuse and redesign.

Computational Environment

The group has 12 SYMBOLICS Lisp Machines, 6 Macintosh-IIs, 2 MacIvories, a Decstation 3100, and 9 Hewlett Packard AI workstations. In addition, the Computer Science Department has a large number of SUN-3/4 workstations, a MIPS, an Intel Hypercube, an Amdahl Hypercube, Evans and Sutherland, Inc.-ES1 Shared Memory Supercomputer, a NeXT machine, Myrias Computer Cooperation-SPS Memory Supercomputer. The department also shares a connection machine with NCAR and UCD. All of these machines are linked together in a network.

Integration and Cooperation

The group is part of the Department of Computer Science. It is a member of the Institute of Cognitive Science at the University of Colorado and is also part of the Intelligent Systems Group (an association of a large number of different institutions at the University of Colorado which are interested in artificial intelligence research). It participates in the Human-Computer Interaction Consortium (which brings together research groups from the USA and Europe working in HCI). It cooperates and has active research relationships with other universities, research laboratories, and industry.

Financial Support

The research is supported by grants and contributions from the following organizations:

1. Current: National Science Foundation (NSF), Army Research Institute (ARI), NYNEX (White Plains, NY), Software Research Associates (Boulder, CO and Tokyo), Apple (Cupertino, CA), and HP (Palo Alto, CA)
2. Previous: Office of Naval Research (ONR), Colorado Institute for Artificial Intelligence, U S WEST Advanced Technologies (Denver, CO), AT&T (Denver, CO), MCC (Austin, TX), NCR (Fort Collins, CO), and Symbolics (Cambridge, MA)

Attached are:

- **Appendix 1:** List of Publications
- **Appendix 2:** List of Technical Reports
- **Appendix 3:** Short Summaries of Major Research Grants
- **Appendix 4:** Screen Images of Some of our Systems

Copies of Publications, Technical Reports, and a list of all publications (including those prior to 1988) can be obtained by contacting:

Francesca Iovine
Department of Computer Science
Campus Box 430
University of Colorado
Boulder, CO 80309-0430

phone: (303) 492-1592
email: iovine@boulder.colorado.edu
FAX: (303) 492-2844

Appendix 1: List of Publications: 1988 - present

1. G. Fischer: **Communication Requirements for Cooperative Problem Solving Systems**, Special Issue on "Knowledge Engineering" of the International Journal "Information Systems" (in press).

   Despite lip service that "most knowledge-based systems are intended to be of assistance to human endeavor and are almost never intended to be autonomous agents," knowledge-based systems research has not been focused enough on the nature and the requirements of cooperative problem solving systems. The emphasis of our work is on creating computer systems to facilitate the cooperation between a human and a computer. Cooperation requires more from a system than having a nice user interface or supporting natural language dialogs. One needs a richer theory of problem solving, which analyzes the functions of shared representations, mixed-initiative dialogues, argumentation, and management of trouble. Our evolving theoretical framework for this approach has led to a number of prototypical systems developments which serve as vehicles for future research. Examination of these systems provides evidence that learning and effective problem solving can be improved through the use of cooperative problem solving systems.

2. G. Fischer, A.C. Lemke, T. Mastaglio, A. Morch: **Critics: An Emerging Approach to Knowledge-Based Human Computer Interaction**, Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA) ACM, New York, April 1990 (in press).

   We describe the critiquing approach to building knowledge-based interactive systems. Critiquing supports computer users in *their* problem solving and learning activities. The challenges for the next generation of knowledge-based systems provide a context for the development of this paradigm. We discuss critics from the perspective of overcoming the problems of high-functionality computer systems, of providing a new class of systems to support learning, of extending applications oriented construction kits to design environments, and of providing an alternative to traditional autonomous expert systems. One of the critiquing systems we have built -- JANUS, a critic for kitchen design -- is described as a reference point for presenting the key aspects of the critiquing process. We then survey additional critiquing systems developed in our and other research groups. The paper concludes with a discussion of experiences and extensions to the paradigm.

3. G. Fischer, A. Girgensohn: **End-User Modifiability in Design Environments**, Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA) ACM, New York, April 1990 (in press).

   Convivial systems encourage users to be actively engaged in generating creative extensions to the artifacts given to them. They have the potential to break down the counterproductive borderline between programming and using programs. Knowledge-based design environments are prototypes for convivial systems. They support human problem-domain communication allowing users to work within their domains of expertise. One of the design rationale behind design environments is to ease the construction and modification of artifacts designed *within* the environment. But because design environments are intentionally no general purpose programming environments, situations will arise that require that the design environment itself is modified. The rationale and the techniques for these later modifications are discussed in this paper. Our conceptual framework for end-user modifiability is illustrated in the context of JANUS, an environment for architectural design. Evaluating our system building efforts against our objectives shows the subtleties of integrating end-user modifiability in these kind of systems.

4. G. Fischer, T. Mastaglio, B. Reeves, J. Rieman: **Minimalist Explanations in Knowledge-Based Systems**, Proceedings of the 23rd Hawaii International Conference on Systems Sciences, January 1990 (in press).

   Research in discourse comprehension and human-computer interaction indicates that good explanations are usually brief. A system that provides brief explanations, however, must plan for the case where brevity comes

at the expense of understanding. Human to human dialog is, to a large part, concerned with conversational repair and question-answer episodes; computer systems need to provide similar fallback techniques to their users. We have designed such an explanation system in the context of a knowledge-based critiquing system, LISP. The system provides several levels of explanations, specifically tailored to the user. If the initial, brief explanation is insufficient, the system positions the user at an appropriate point within a more complete, hypertext-based documentation system. Rather than attempting to design a system that can generate a perfect, one-shot explanation for any given situation, this approach concentrates on matching the communication abilities provided by current computer technology to the cognitive needs of the human user.

5. T. Mastaglio: **User Modeling in Computer-Based Critics**, Proceedings of the 23rd Hawaii International Conference on Systems Sciences, January 1990 (in press).

A user modeling approach for computer-based critics is described. The critiquing model is one approach to instantiating the cooperative problem solving paradigm. The theoretical background for cooperative problem solving and the motivation for developing a user modeling approach in this domain are based on a need to provide systems that operate independent of explicit user direction. A theoretical model of user domain knowledge, an analysis of the application domain, LISP, and research on the generation of explanations prescribe the contents of the user model. How to represent, acquire, and maintain consistency of the user model in a critiquing system for LISP programming -- LISP -- is the fundamental issue addressed. LISP has been extended to include a user modeling component; this component includes a database of information about the user and a modeling agent. The modelling agent encapsulates the access and update methods for the user model.

6. S. Doane, A.C. Lemke: **Using Cognitive Simulation to Develop User Interface Design Principles**, Proceedings of the 23rd Hawaii International Conference on Systems Sciences, January 1990 (in press).

This paper summarizes some of our current research using simulation of user performance to develop theory-based user interface design principles. The approach is to develop independent detailed simulations of user command production in two high-functionality systems: the UNIX operating system and FRAMER, an innovative user-interface design tool. The simulations are based on the construction-integration theory and build on the action planning model developed by Mannes and Kintsch. Use of the model provides some insight into why certain types of command productions pose problems for UNIX users, and it shows promise for explaining problems for users of FRAMER. We discuss using these insights as tools for developing user interface design principles for future system design.

7. G. Fischer: **The Importance of Models in Making Complex Systems Comprehensible**, Proceedings of the 8th Interdisciplinary Workshop on Informatics and Pyschology, Schaerding, North Holland, 1990 (in press).

Reality is not user-friendly. To cope, model, and comprehend a complex reality requires complex systems. Complex systems offer power, but they are not without problems. High-functionality computer system serve as prototypical examples for complex systems in our research. They are used to instantiate our model of cooperative problem solving in joint human-computer systems. Models play a crucial part in creating these systems and in increasing their comprehensibility. Three different types of models are discussed in this paper: the designers' models of the users, the users' models of the systems, and the systems' models of the users. Innovative system designs supporting human-problem domain communication and providing design environments illustrate the relevance of these models in making complex systems comprehensible.

8. G. Fischer: **Creativity Enhancing Design Environments**, Proceedings of the International Conference "Modeling Creativity and Knowledge-Based Creative Design," Heron Island, Australia, December 1989 (in press).

Computers have the potential to be creativity enhancing tools. But most of the current systems have not lived up to these expectations -- they have restricted rather than enhanced creativity. Designers were forced to express their goals, ideas, and (partial) solutions at levels that were too remote from the problem domains they were dealing with. To overcome these limitations, we have developed a conceptual framework and prototypical systems that allow designers to work with personal meaningful operations. Beyond providing domain-specific abstractions, our knowledge-based design environments can evaluate and criticize an evolving design and provide feedback to the designer. They integrate constructive and argumentative components. They turn the computer into an invisible instrument and support cooperative problem solving between the human designer and the computer. Our experience with these systems demonstrates that they have the potential to serve as important stepping stones towards creativity enhancing environments.

9. G. Fischer, R. McCall, A. Morch: **JANUS: Integrating Hypertext with a Knowledge-Based Design Environment**, Proceedings of Hypertext'89, November 1989 (in press).

Hypertext systems and other complex information stores offer little or no guidance in helping users find information useful for activities they are currently engaged in. Most users are not interested in exploring hypertext information spaces *per se* but rather in obtaining information to solve problems or accomplish tasks. As a step towards this we have developed the JANUS design environment. JANUS allows designers to construct artifacts in the domain of architectural design and at the same time to be informed about principles of design and the reasoning underlying them. This process integrates two design activities: construction and argumentation. Construction is supported by a knowledge-based graphical design environment, and argumentation is supported by a hypertext system. Our empirical evaluations of JANUS and its predecessors has shown that integrated support for construction and argumentation is necessary for full supprt of design.

10. H.-D. Boecker, G. Fischer, H. Nieper-Lemke: **The Role of Visual Representations in Understanding Software**, in D. Partridge (ed.), "Artificial Intelligence and Software Engineering," Ablex Publishing Corporation, Norwood, NJ, 1989 (in press).

The way a problem is represented strongly affects our ability to understand and solve it. Visual representations are especially important because the human visual system is such a powerful way of processing information. However few existing systems try to take advantage of these insights. In pursuit of the long-range goal of constructing a *software oscilloscope* that makes the invisible visible, we have constructed system components which automatically generate graphical representations of complex structures, illustrate the control flow of complex programs, and support visualization techniques in object-oriented environments. Our tools are used in a variety of contexts: in programming environments, as components in intelligent support systems, and in human-computer interaction in general. Visual representation alone, however, is not enough; the designer of visualization tools must take into account the semantics of graphical symbols and the user's need to limit visualization to the *relevant* facts and relations.

11. G. Fischer, T. Mastaglio, J. Rieman: **User Modeling in Critics Based on a Study of Human Experts**, Proceedings of the Fourth Annual Rocky Mountain Conference on Artificial Intelligence, RMSAI, Denver, CO, June 1989, pp. 217-225.

Computer-based critics are an effective approach for using knowledge-based systems to support cooperative problem solving but need to be extended with user modeling capabilities. Efforts to do this in the LISP-CRITIC system using statistical methods indicated the need to pursue additional techniques for implicit acquisition of knowledge about the user. A verbal protocol study of human experts analyzing the work of other programmers was conducted. The study focused on how these experts infer the knowledge and expertise levels of anonymous programmers when provided only with samples of the programmers' LISP code. Three distinct categories of "cues" to a programmer's knowledge were found: syntactic, code semantic, and problem seman-

tic. Analysis of these categories indicates that the first two are amenable to acquisition by the LISP-CRITIC system, but that the third category requires a significantly different knowledge base than the system currently contains. Knowledge of the world in general and specific problem domains is required. The results of current efforts to incorporate some of these techniques into the LISP-CRITIC are presented.

12. R. McCall, G. Fischer: **Supporting Reflection-In-Action in the JANUS Design Environment**, Proceedings of the CAAD Futures'89 Conference "Computer Aided Design Education" (Pre-Publication Edition), June 1989.

Computer-supported construction facilitates design. The problem is that it facilitates both good and bad design. To create good design, designers need knowledge about how to evaluate what is constructed. To increase the likelihood of good design, a computer-aided design system should supplement support for construction with a store of evaluative knowledge. But adding such knowledge to a construction support system would not in itself be enough. Good design requires that designers have more than knowledge. It requires *reflection-in-action*, i.e., think about what one does while this thinking can still make a difference to what one does. The paper describes an architecture and a prototype of a system which supports *reflection-in-action*.

13. G. Fischer, R. McCall, A. Morch: **Design Environments for Constructive and Argumentative Design**, Human Factors in Computing Systems, CHI'89 Conference Proceedings (Austin, TX), ACM, New York, May 1989, pp. 269-275.

Design Environments are computer systems which support design by enabling cooperative problem solving between designer and computer. There are two complementary problem solving activities in design: constructive design and argumentative design. We have created two computer-supported environments, CRACK and VIEWPOINTS, to support these two activities. CRACK is a knowledge-based critic which has knowledge about how kitchen appliances can be assembled into functional kitchens. VIEWPOINTS is a hypertext system based on the IBIS design methodology and contains useful information about the principles of kitchen design. The integration of these two types of systems will eliminate shortcomings of the individual systems.

14. G. Fischer, H. Nieper-Lemke: **HELGON: Extending the Retrieval by Reformulation Paradigm**, Human Factors in Computing Systems, CHI'89 Conference Proceedings (Austin, TX), ACM, New York, May 1989, pp. 357-362.

People who attempt to use a complex information store on a computer encounter a number of problems: They do not know what information exists or how to find information, they get no support in articulating a question, and they are unable to phrase their question in terms that the system understands. HELGON, an intelligent environment that supports limited cooperative problem solving, helps people deal with complex information stores. HELGON supports retrieval and editing by reformulation with multiple specification techniques, and it acquaints the user with the system model of the information store. Within the current HELGON system, a number of different information stores have been implemented. Empirical evaluations have shown that HELGON supports effective communication. In addition, the evaluations have shown interesting extensions for future work.

15. G. Fischer: **Human-Computer Interaction Software: Lessons Learned, Challenges Ahead**, IEEE Software, Vol. 6, No. 1, January 1989, pp. 44-52.

Human-computer interaction (HCI) is an ill-structured domain that is limited much more by specification than by implementation. When we write software for HCI, we define not only what the computer will do but also what humans will and can do -- and we make assumptions about what they want to do. Humans are individuals. They have different knowledge and different preferences, and they change; that is, a novice may become an expert over time. In our research over the last ten years, we have tried to improve human-computer interaction by widening the explicit communication channel between humans and computers and by establishing with knowledge-based systems an implicit communication channel in which shared knowledge is the basis for cooperative problem solving. Our efforts have taught us a number of lessons such as the limitations of user

interface toolkits, the need for higher level, application-oriented abstractions, and the need for intelligent support systems. We have defined several challenges for the future: the need for methodologies and tools for coping with design tasks with incomplete specifications, the challenge of resolving design trade-offs, and the need to support cooperative problem solving.

16. G. Fischer, T. Mastaglio: **Computer-Based Critics**, Proceedings of the 22nd Hawaii Conference on System Sciences, Vol. III: Decision Support and Knowledge Based Systems Track, IEEE Computer Society, January 1989, pp. 427-436.

    Computer-based critics are a paradigm for intelligent human-computer communication which overcomes a number of limitations of other approaches, such as tutoring and advising. Critics are much more user-centered and support users in their own doing. They provide information only when it is relevant. They allow users to do whatever they want and interrupt only when users' plans, actions, or products are considered significantly inferior. They are applicable in situations where users have some basic competence in carrying out a task, because users must be able to generate a plan, action, or product by themselves. They are most useful in domains where no unique best solution exists but where trade-offs have to be carefully balanced. Critics need to be knowledge-based systems. Critic systems must incorporate knowledge of the application domain, support dynamic explanation generation, model individual users, and provide innovative user interfaces. Over the last few years, we have implemented a number of critics in different domains (e.g., for programming, for design). The rationale, the design, and the evaluation of these systems is described as a starting point towards a general framework for computer-based critics.

17. S. Henninger, A. Ignatowsky, C. Rathke, D. Redmiles: **A Knowledge-Based Design Environment for Graphical Network Editors**, Proceedings of the 22nd Hawaii Conference on System Sciences, Vol. II: Software Track, IEEE Computer Society, January 1989, pp. 881-891.

    Design systems for graphical network editors are general purpose tools that capture common characteristics of network-like structures. As a consequence, these systems support their users only as far as the network functionality is concerned. While this is important, it is not enough to effectively support users in designing specific network viewers or editors for specific applications. Without knowledge about the application domain, for which the editor is designed, there is little potential to guide the user or make suggestions for a better design. We have developed a graphical editor design environment that incorporates and applies knowledge about application domains. Our goal is to boost a design environment closer towards its application. As an example of this new generation of design support systems, we have developed a design environment for graphical editors in the domain of object-oriented inheritance networks. In addition to the general knowledge about graphs, the system knows about inheritance mechanisms in object-oriented systems, it knows about the nodes being classes and the links representing the superclass relation. This knowledge is used to provide guidance, critique, and constraints.

18. G. Fischer: **Cooperative Problem Solving Systems**, Proceedings of the 1st Simposium Internacional de Inteligencia Artificial (Monterrey, Mexico), October 1988, pp. 127-132.

    Problem solving effectiveness is often enhanced by cooperation -- traditionally, cooperation among people, or, more recently, cooperation between a human and a computer. The emphasis of our work is on creating computer systems to facilitate the cooperation between a human and a computer. Examination of these systems provides evidence that learning and effective problem solving can be improved through the use of cooperative systems. It also indicates the need for a richer theory of problem solving, which would analyze the functions of shared representations, mixed-initiative dialogues, and management of trouble, if problems occur.

19. G. Fischer, C. Rathke: **Knowledge-Based Spreadsheet Systems**, Proceedings of AAAI-88, Seventh National Conference on Artificial Intelligence (St. Paul, MN), Morgan Kaufmann Publishers, San Mateo, CA, August 1988, pp. 802-807.

    Spreadsheet systems have changed the way the world perceives and deals with computers. In an attempt at maintaining the positive elements of spreadsheets while overcoming some of their limitations, we have developed FINANZ, a computational environment to develop financial planning systems. FINANZ contains a form-based user interface construction system, which allows the creation of advanced user interfaces without the need for conventional programming. It uses constraint-based programming for the representation of knowledge about the application domain. Its layered architecture (based on object-oriented knowledge representation) supports the modification and extension of the system and the dynamic generation of explanations.

20. G. Fischer, A. Morch: **CRACK: A Critiquing Approach to Cooperative Kitchen Design**, Proceedings of the International Conference on Intelligent Tutoring Systems (Montreal, Canada), June 1988, pp. 176-185.

    Human problem-domain communication and cooperative problem solving are two enabling conditions that allow users, who are not computer experts, to use computers for their own purposes. Computer-based critics, a specific class of intelligent support systems, are most effective if they are embedded in a framework defined by human problem-domain communication and cooperative problem solving. CRACK is a specific critic system which supports users designing kitchens. It provides a set of domain-specific building blocks and has knowledge about how to combine these building blocks into useful designs. It uses this knowledge "to look over the shoulder" of a user carrying out a specific design. If CRACK, based on its understanding of kitchen design, discovers a shortcoming in users' designs, it offers criticism, suggestions, and explanations and assists users in improving their designs through a cooperative problem solving process. CRACK is not an expert system that dominates the design process by generating new designs from high-level goals or resolving design conflicts automatically. Users control the behavior of the system at all times (e.g., the critiquing can be "turned on and off"), and if users disagree with CRACK, they can modify its knowledge base.

21. G. Fischer, S.A. Weyer, W.P. Jones, A.C. Kay, W. Kintsch, R.H. Trigg: **A Critical Assessment of Hypertext Systems**, Human Factors in Computing Systems, CHI'88 Conference Proceedings (Washington, D.C.), ACM, New York, May 1988, pp. 223-227.

    Over forty years ago, Vannevar Bush articulated his vision of a "Memex" machine: *"associative indexing, ... whereby any item may be caused at will to select immediately and automatically another."* In the sixties, Engelbart built collaborative systems to provide idea structuring and sharing. Nelson coined "hypertext" and proposed world-wide networks for publishing, linking, annotating, and indexi ng multiple versions of documents. With increasing numbers of research projects, papers, panels, and conferences, and commercially available systems (e.g., NOTECARDS by Xerox, GUIDE by Owl and HYPERCARD by Apple) in recent years, hypertext may be an idea whose time has finally come -- or at least a phenomenon not to be ignored.

22. G. Fischer: **Enhancing Incremental Learning Processes with Knowledge-Based Systems**, in A. Lesgold and H. Mandl (eds.), "Learning Issues for Intelligent Tutoring Systems," Springer Verlag, Berlin - Heidelberg - New York, 1988, pp. 138-163, Ch. 7.

    Over the last several years we have developed aspects of a general theory of incremental learning processes which will be the dominant way to master systems of broad functionality. To complement the theoretical work we are pursuing the goal of building a LISP-CRITIC and we have constructed several systems to be used as its components. All these systems have two things in common: they are knowledge-based and use innovative techniques to enhance human-computer communication.

23. G. Fischer, A.C. Lemke: **Constrained Design Processes: Steps Towards Convivial Computing**, in R. Guindon (ed.), "Cognitive Science and its Application for Human-Computer Interaction," Lawrence Erlbaum Associates, Hillsdale, NJ, 1988, pp. 1-58.

    Our goal is to construct components of convivial computer systems which give people who use them the greatest opportunity to enrich their environments with the fruits of their vision. Constrained design processes are a means of resolving the conflict between the generality, power and rich functionality of modern computer systems, and the limited time and effort which casual and intermediate users want to spend to solve their problems without becoming computer experts. Intelligent support systems are components which make it less difficult to learn and use complex computer systems. We have constructed a variety of design kits as instances of intelligent user support systems which allow users to carry out constrained design processes and give them control over their environment. Our experience in building and using these design kits will be described.

24. G. Fischer, A.C. Lemke: **Construction Kits and Design Environments: Steps Toward Human Problem-Domain Communication**, Human-Computer Interaction, Vol. 3, No. 3, 1988, pp. 179-222.

    Our goal is to build cooperative computer systems to augment human intelligence. In these systems the communication between the user and the computer plays a crucial role. To provide the user with the appropriate level of control and a better understanding, we have to replace human-computer communication with *human problem-domain communication*, which allows users to concentrate on the problems of their domain and to ignore the fact that they are using a computer tool. Construction kits and design environments are tools that represent steps towards human problem-domain communication. A construction kit is a set of building blocks that models a problem domain. The building blocks define a design space (the set of all possible designs that can be created by combining these blocks). Design environments go beyond construction kits in that they bring to bear general knowledge about design (e.g., which meaningful artifacts can be constructed, how and which blocks can be combined with each other) that is useful for the designer. Prototypical examples of these systems (especially in the area of user interface design) are described in detail and the feasibility of this approach is evaluated.

Appendix 2: List of Technical Reports and Papers in Preparation

1. G. Fischer, A.C. Lemke: **Knowledge-Based Design Environments for User Interface Design**, Technical Report, Department of Computer Science, Boulder, CO, 1989.

2. A.C. Lemke: **Design Environments for High-Functionality Computer Systems**, PhD Thesis, Department of Computer Science, University of Colorado, Boulder, CO, July 1989.

3. S. Henninger: **CODEFINDER: Using Associative Networks and Spreading Activation for Software Object Retrieval**, Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, April 1989.

4. G. Fischer, P. Foltz, W. Kintsch, H. Nieper-Lemke, C. Stevens: **Personal Information Systems and Models of Human Memory**, Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, April 1989.

5. G. Fischer, W. Kintsch, P.W. Foltz, S.M. Mannes, H. Nieper-Lemke, C. Stevens: **Theories, Methods, and Tools for the Design of User-Centered Computer Systems** (Interim Project Report, September 1986 - February 1989), Department of Computer Science, University of Colorado, Boulder, CO, March 1989.

6. G. Fischer, T. Mastaglio: **A Conceptual Framework for Knowledge-Based Critic Systems**, Submitted to the International Journal "Decision Support Systems," North Holland.

7. G. Fischer, A.C. Lemke: **FRAMER: Integrating Working and Learning**, Technical Report, Department of Computer Science, University of Colorado, Boulder, CO, December 1988.

8. G. Fischer, A.C. Lemke, H. Nieper-Lemke: **Enhancing Incremental Learning Processes with Knowledge-Based Systems (Final Project Report)**, Technical Report No. CU-CS-392-88, Department of Computer Science, University of Colorado, Boulder, CO, March 1988.

9. T.T. Turner (ed): **Mental Models and User-Centered Design**, Workshop Report (Breckenridge, CO), Technical Report No. 88-9, Institute of Cognitive Science, University of Colorado, Boulder, CO, 1988.

10. G. Fischer, H. Nieper: **Personalized Intelligent Information Systems**, Workshop Report (Breckenridge, CO), Institute of Cognitive Science, University of Colorado, Boulder, CO, Technical Report No. 87-9, 1987.

Appendix 3: Short Summaries of Current Major Research Grants

1. Theories, Methods, and Tools for the Design of User-Centered Computer Systems

Sponsor: Army Research Institute

Principal Investigators: Gerhard Fischer, Walter Kintsch

Period of Support: August 1986 - July 1991

Project Summary:

The goal of this research at the general level is to develop *theories, methods, and tools for the design of user-centered computer systems* and at the specific level to design, implement and evaluate a customizable *Personalized Intelligent Retrieval System (PIRS)*. Our research effort is based on the basic hypothesis that the following duality exists:

1. User-centered system design cannot be done and understood without trying to test existing ones, extend existing ones, and design new ones.
2. User-centered system design cannot be understood by just doing it; the system building efforts must be based on a deep understanding of the theoretical and methodological issues behind them, derived primarily from Cognitive Science and as far as evaluation is concerned from Human Factors / Cognitive Ergonomics.

The software systems developed within our research effort should be significant not only as technical achievements in computer science, but also because these systems are based upon *principled* analyses of how one can best help humans to cope with complex information systems. If one wants to accomplish good work in user-centered human-computer systems, it is not sufficient to know only how to build these systems: one must also be prepared to discover *which systems are worth building* and on which *principled design strategies* these systems can be based. Only joint expertise in computer science, cognitive science and human factors will make this possible.

The *expected results* of this long-term research effort will be substantial contributions to a number of significant scientific and practical problems:

1. the theory of design and comprehension of complex user-centered systems,
2. the theory of knowledge use and knowledge retrieval for designers at all levels,
3. the definition, exploration and instantiation of new methodologies (e.g., evolutionary design methodologies, coevolution between specifications and implementations to cope with design instabilities, strategies to incrementally derive well-structured problems from ill-structured ones, end-user control over systems),
4. the design, implementation and evaluation of a customizable PIRS,
5. a set of well-tested tools and metatools organized in a PIRS,
6. guidelines for constructing user-centered systems based on the above theories, methodologies, tools, and metatools,
7. quantitative and qualitative measures to be used by designers and Human Factors specialists to evaluate the design of user-centered systems.

The proposed research builds upon many years of psychological research on text comprehension and the development of advanced computer systems over the past 8 years to enhance human-computer communication with knowledge-based systems.

2. Design Principles for Comprehensible Systems

Sponsor: National Science Foundation

Principal Investigators: Gerhard Fischer, Walter Kintsch, Clayton Lewis, Peter Polson

Period of Support: August 1988 - July 1991

Project Summary:

Modern high-function computer systems are difficult to master and use. To attack this problem, we propose a program of research that combines fundamental theoretical work on the cognitive processes involved in computer use with the development of innovative systems that embody new design approaches.

The main objective is to develop design principles for making comprehensible systems: systems that are radically easier for users to understand than current systems. These principles will be developed in the context of two prototype application systems, which will serve as testbeds for the principles as well as illustrations of them. The prototype applications will be the foci of interaction of two lines of work: the exploration of new interface ideas, and the development of a new, comprehension-centered theory of human-computer interaction. These lines of work build on and draw together our previous and current work in exploratory system building and in cognitive theory. Both the prototype applications and the underlying theoretical developments will be evaluated empirically.

This project is unique in bringing together fundamental theoretical work and exploratory system building in tight cooperation. We feel this will pay dividends for both sides of the project. Findings from exploration of new interface ideas will have more general impact if they can be placed on a theoretical basis. Theory developed in conjunction with advanced system design will be more relevant to future technology.

Appendix 4: Screen Images of Some of our Systems

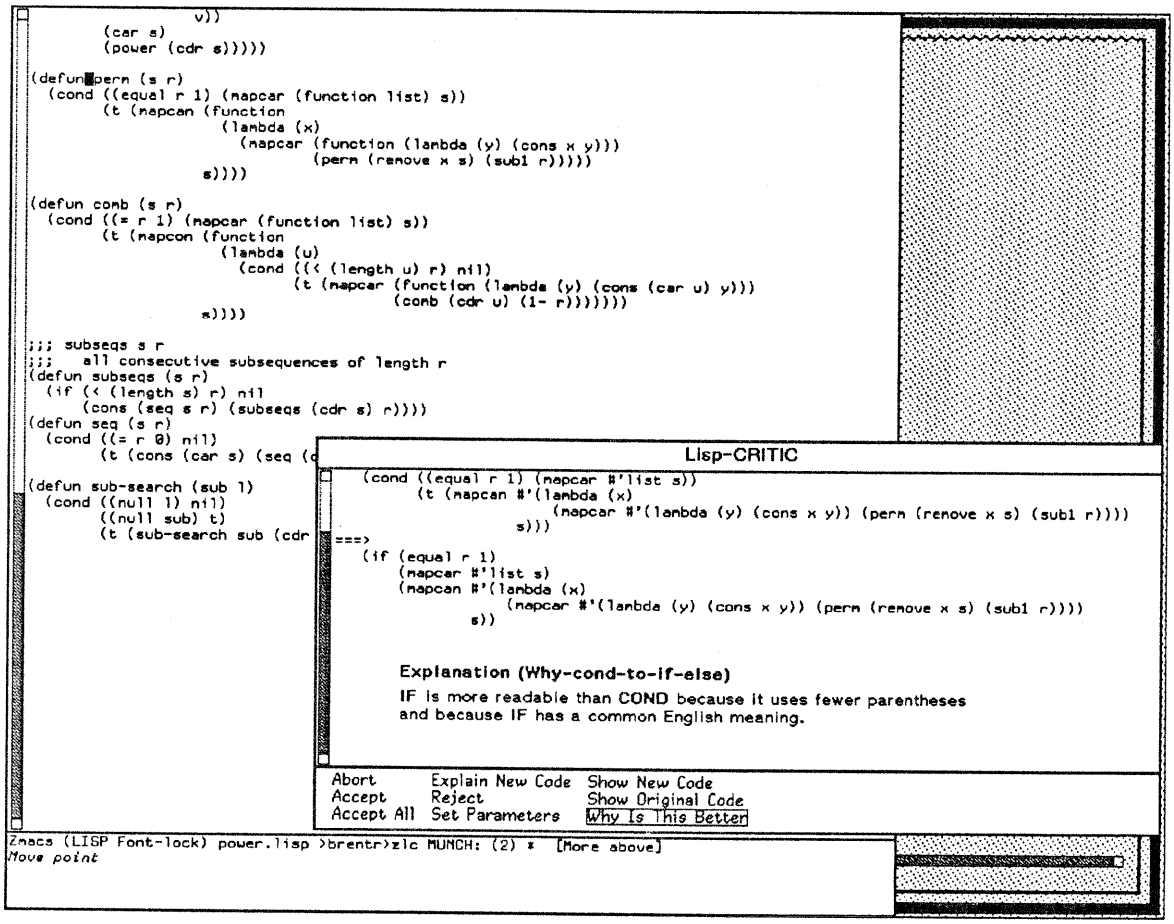

Figure 1: The User Interface of LISP

LISP-CRITIC Is a knowledge-based system that critiques LISP code. The interface shows the user working with LISP-CRITIC. The large editor window contains a program being worked on by the user. When invoked by the user, LISP-CRITIC opens a window in which it makes suggestions to the user about how to improve the program being written and provides explanations of its suggestions when asked. In this figure, the LISP-CRITIC displays a cond-to-if transformation and an explanation of why LISP-CRITIC recommended changing the **cond** function to an **if**.

See publications No. 4, 5, 16, and 22.

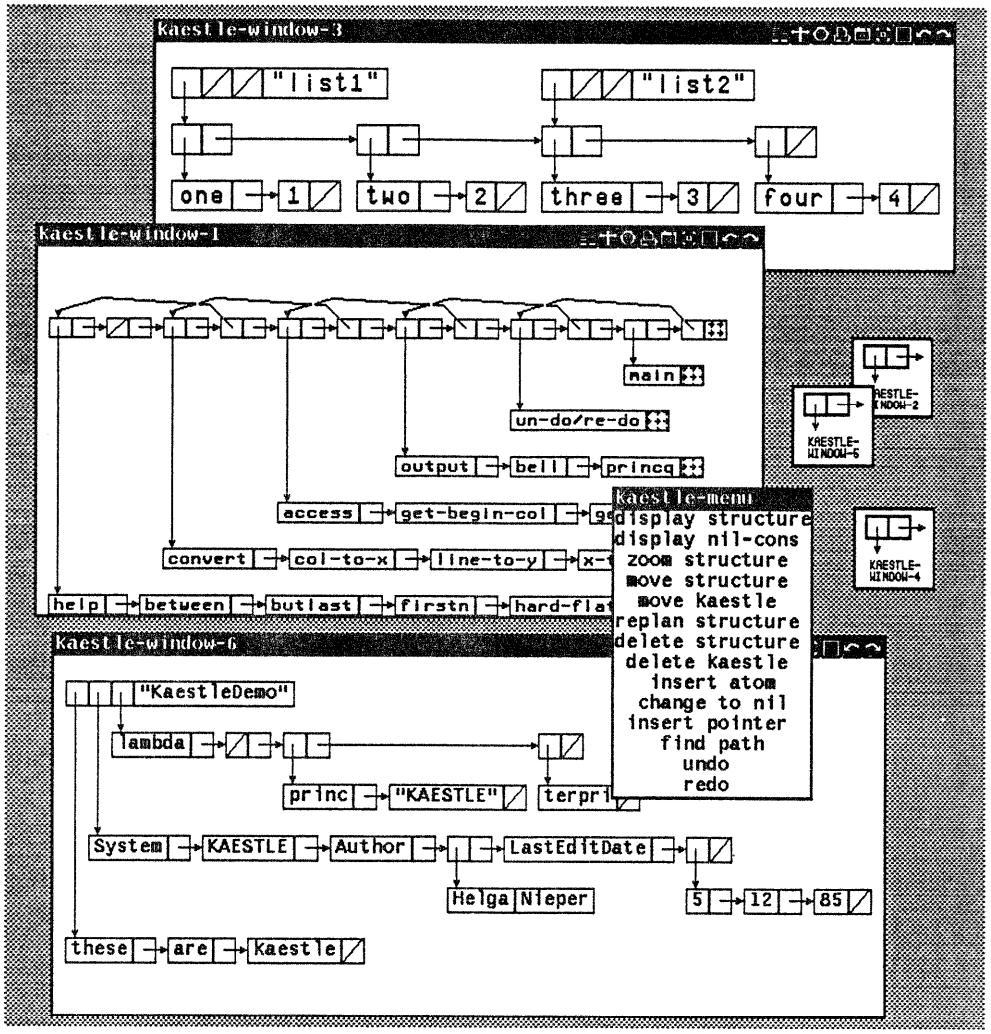

Figure 2: KAESTLE

With KAESTLE, the graphic representation of a LISP data structure is generated automatically and can be edited directly with a pointing device. By editing we do not only mean changing the structure itself but rearranging the graphical layout as well.

See publication No. 10.

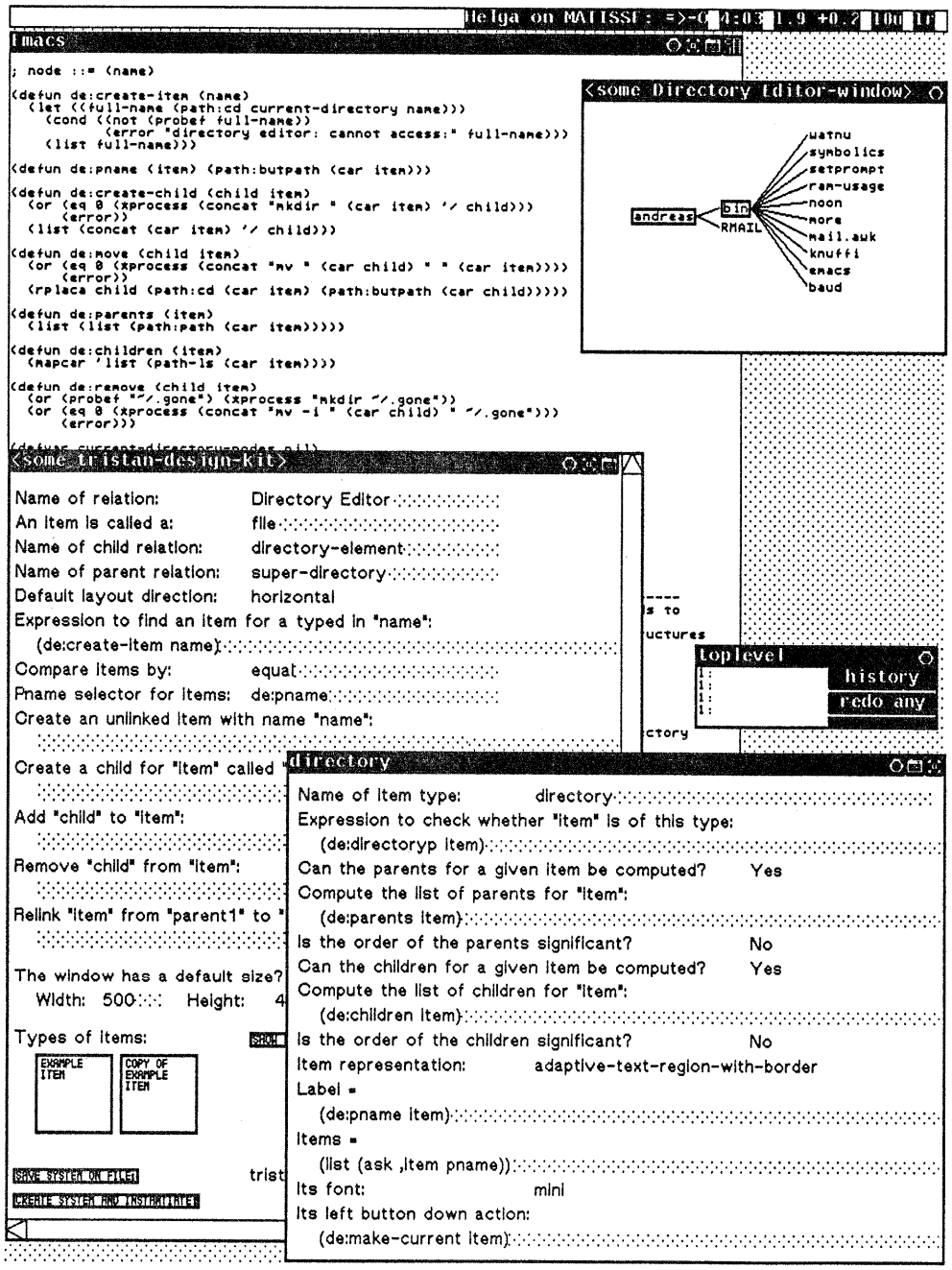

Figure 3: TRIKIT

TRIKIT is a high-level interface to TRISTAN for application programmers. TRISTAN is a generic display and editing system for directed graphs, used for example, in HELGON (see Figure 9) and INFOSCOPE (see Figure 11). TRIKIT hides implementation details of TRISTAN and the implementation language. The forms have to be filled in with calls to the application.

See publications No. 23 and 24.

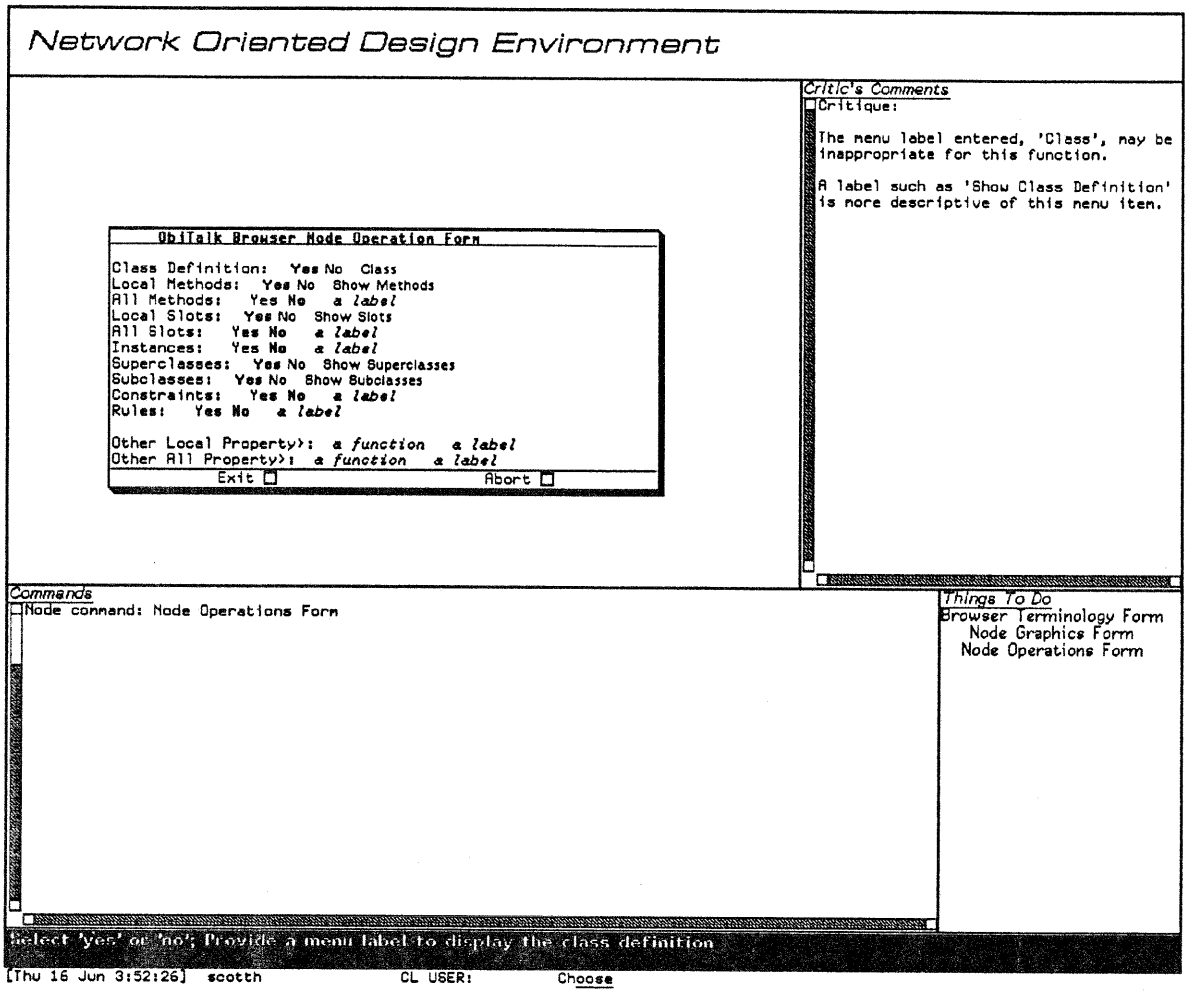

Figure 4: NODE

NODE (Network-Oriented Design Environment) is a design environment for creating user interfaces for programs that interact with network structures. Examples include class inheritance networks in object-oriented languages or layouts of communication networks. The NODE user interface consists of a display pane where input forms are displayed, a listener pane which allows the designer to communicate with the operating system, a command pane through which the user chooses which form to display, and a critic pane, where constraint and critic information is displayed. Note that a critic has responded to the user changing a label to something not very meaningful.

See publication No. 17.

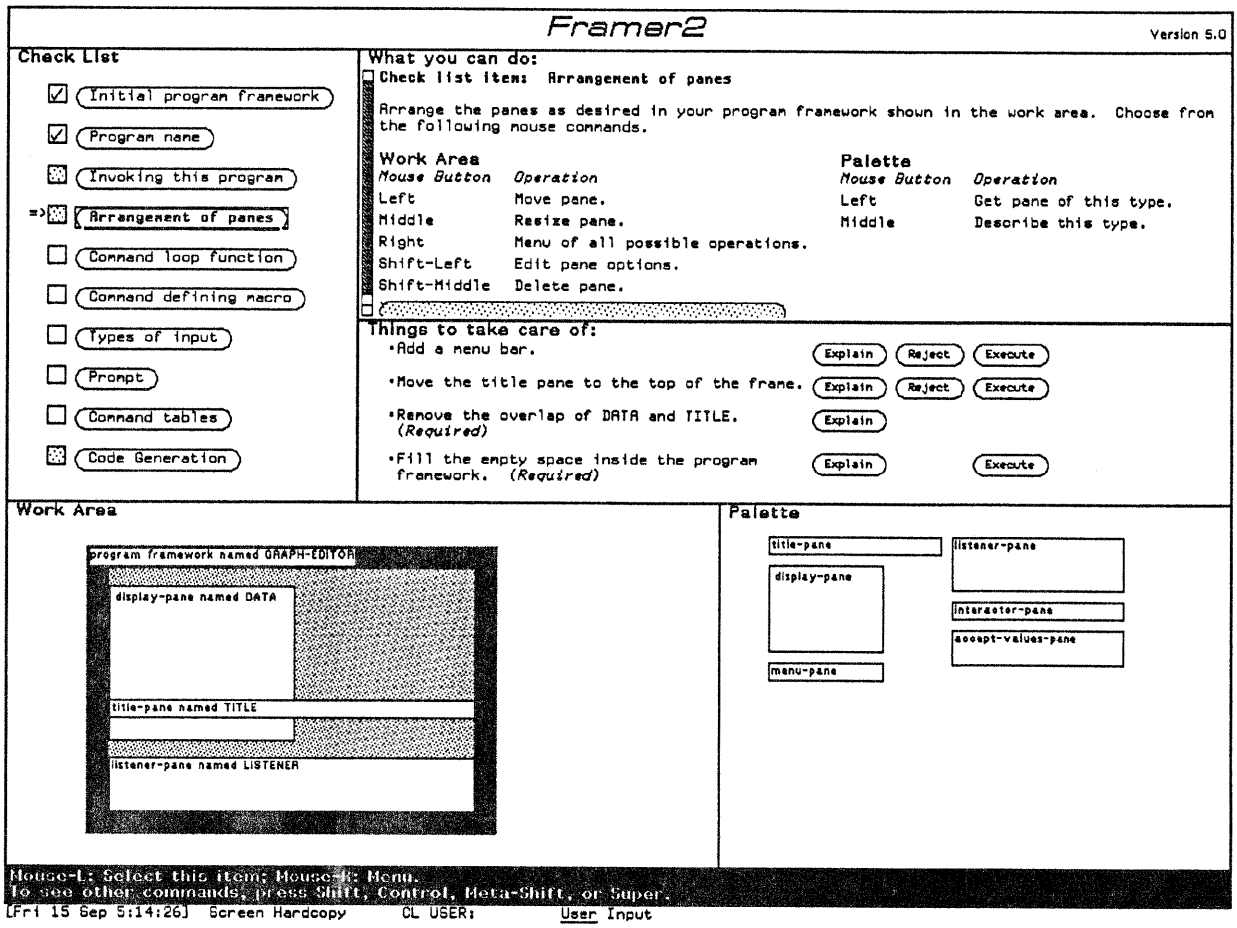

Figure 5: FRAMER

FRAMER is a knowledge-based design environment for window-based user interfaces. FRAMER supports the software design process in multiple ways. It makes components readily available through a graphical direct manipulation interface. FRAMER takes care of interdependencies of components, and FRAMER assesses designs by highlighting good aspects and suggesting improvements to eliminate the shortcomings of poor designs. This last feature is implemented as a rule-based critic that can be activated by the user at any time of the design process. FRAMER translates the graphical representation into executable program code and vice versa. This allows rapid prototyping and transparent modifications at both levels.

See technical reports No. 1 and 2.

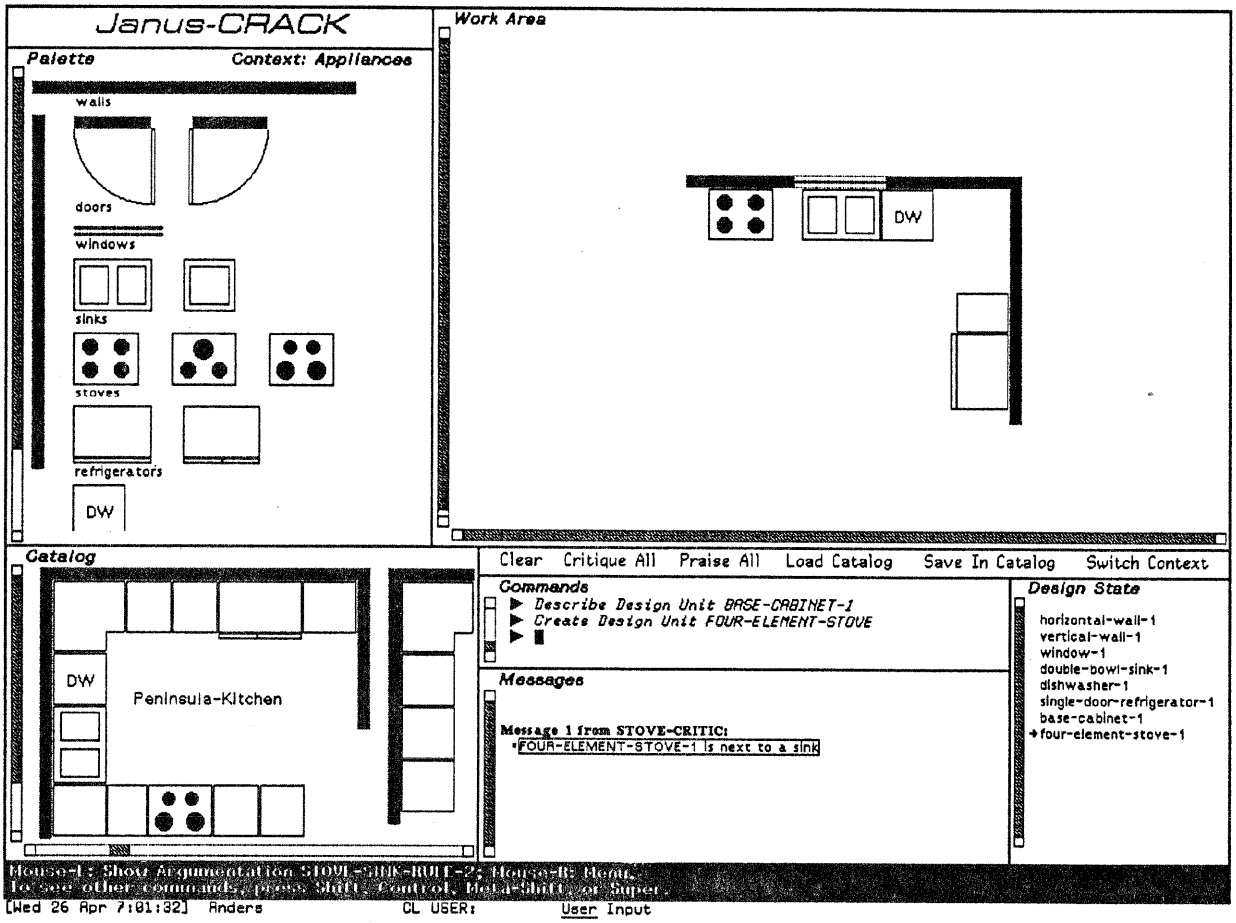

Figure 6: JANUS: The Construction Component

Cooperative problem solving environments for design require support for constructive and argumentative design. The JANUS design environment allows designers to construct architectural floor plan layouts and at the same time to be informed about the general principles underlying such constructions. JANUS, does not try to automate the design process by replacing the human designer with an expert system, but the computer is used as a tool and an agent in cooperation with the designer to enrich traditional design practice. JANUS is both a useful learning environment for design students as well as an efficient tool for skilled designers.

JANUS is the most recent in a series of system building efforts starting with the two systems CRACK and VIEWPOINTS which support constructive and argumentative aspects of design, respectively. CRACK is a knowledge-based design environment which has knowledge about how basic design units can be assembled into functional designs.

See publications No. 9, 12, 13, and 20.

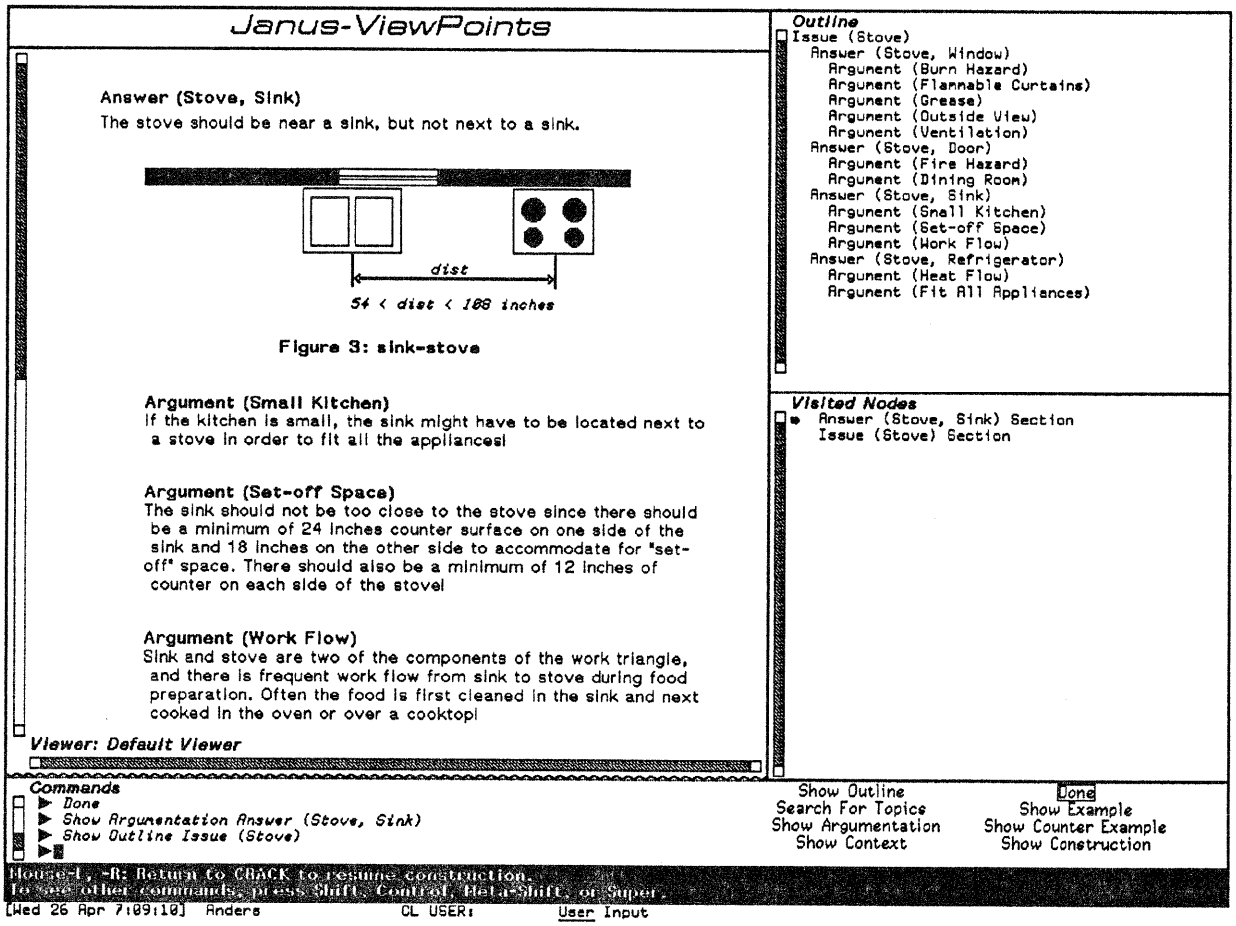

Figure 7: JANUS: The Argumentation Component

VIEWPOINTS is an issue-based hypertext system which contains useful information about general principles of design. Critics link the two systems. Evaluation of the individual systems have identified shortcomings which have lead to the design and implementation of JANUS, an integration of CRACK and VIEWPOINTS. Evaluation of JANUS has shown that the integrated system is able to overcome some of the limitations of the original systems, and that problematic situations designers can come up in during construction can be resolved by argumentation and vice versa.

See publications No. 9, 12, 13, and 20.

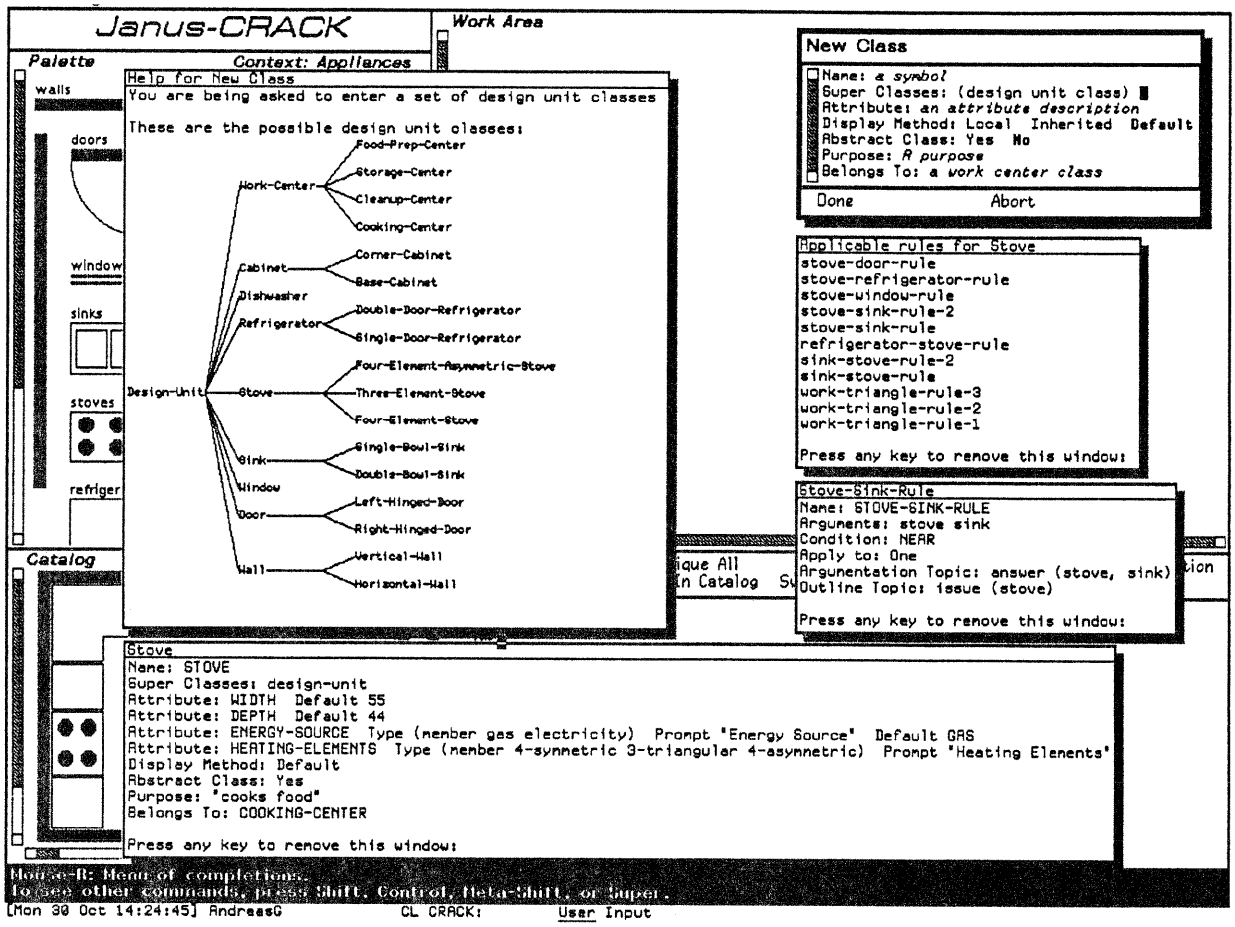

Figure 8: End-User Modifiability in JANUS

End-user modifiability is of crucial importance in knowledge-based design environments, because these systems do not try to serve as general purpose programming environments but provide support for specific tasks. In cases where designers of these environments have not anticipated specific activities, users must be able to modify the design environment itself.

Situations will arise in which users want to design a kitchen with appliances that are not provided by the design environment. Property sheets help users define new design unit classes or modify existing ones by eliminating the need to remember names of attributes. The modification process is supported with context-sensitive help (e.g., showing users constraints for the value of a field).

The system supports the finding of an appropriate place for the new class in the class hierarchy by displaying the current hierarchy. The user can display the definition of every class in the hierarchy with a mouse click. In addition, the applicable critic rules for a class can be listed, and the definition of each of these rules can be be displayed.

See publication No. 3.

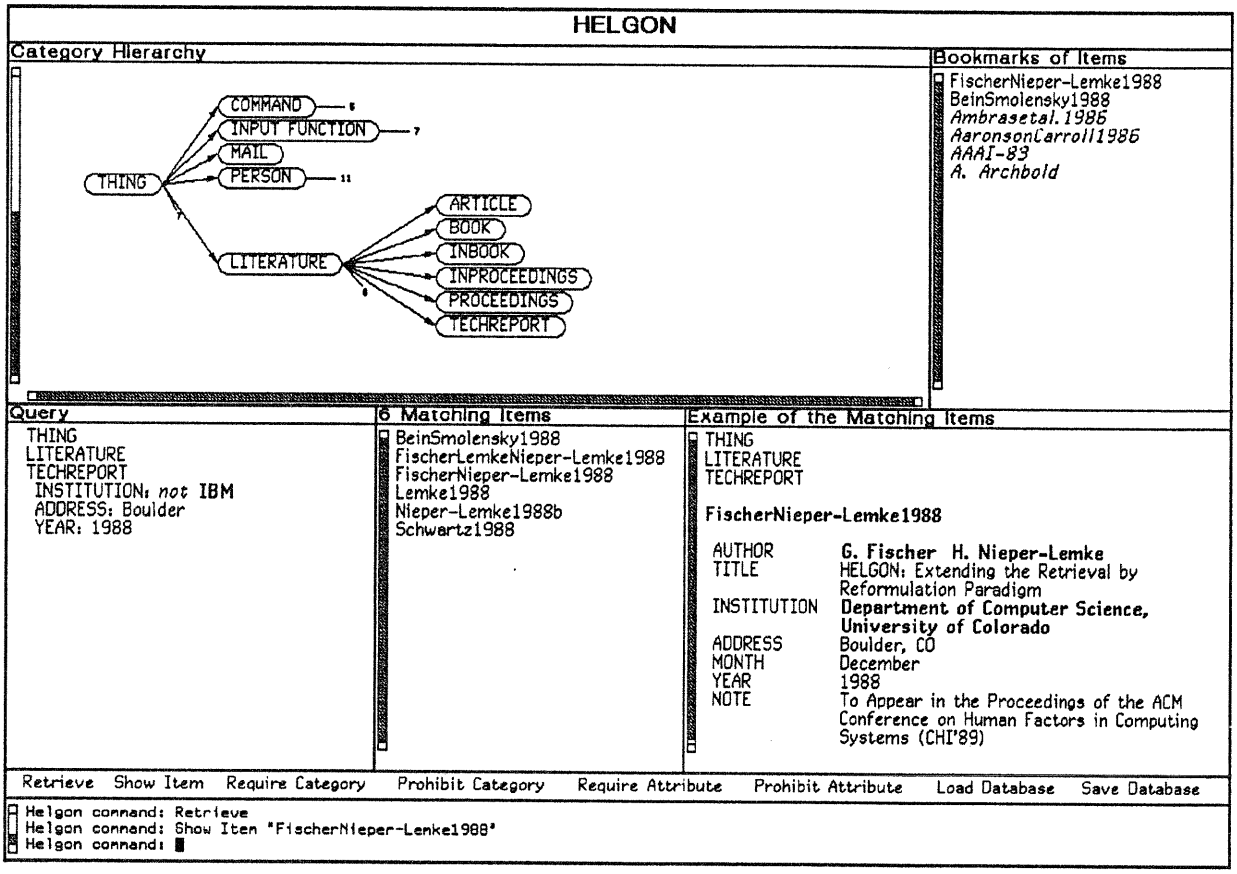

Figure 9: HELGON

HELGON is an information system that is based on the *retrieval by reformulation* paradigm. A query is created incrementally. It consists of categories and attribute restrictions associated with the categories (see the **Query** pane). The set of items matching the current query is shown in the **Matching Items** pane; one example item is shown in the **Example of the Matching Items** pane. A graphical display of the hierarchical classification of the information store is shown in the **Category Hierarchy** pane. Categories as well as attribute values can be either "required" or "prohibited." Users can do this by selecting them from the information displayed or from a menu of alternative values. Thus users who are not familiar with the knowledge base or who do not have a well-specified goal can be guided towards the appropriate information. On the other hand, users who know exactly what they want can add categories or attribute restrictions to the query directly through input from the keyboard. In this case, only existing values are accepted as input, partial input is completed automatically if possible, and users can get a list of all current possibilities. HELGON also allows users to *edit* information and integrates the addition of new information with the retrieval by reformulation paradigm.

See publication No. 14 and technical reports No. 4 and 5.

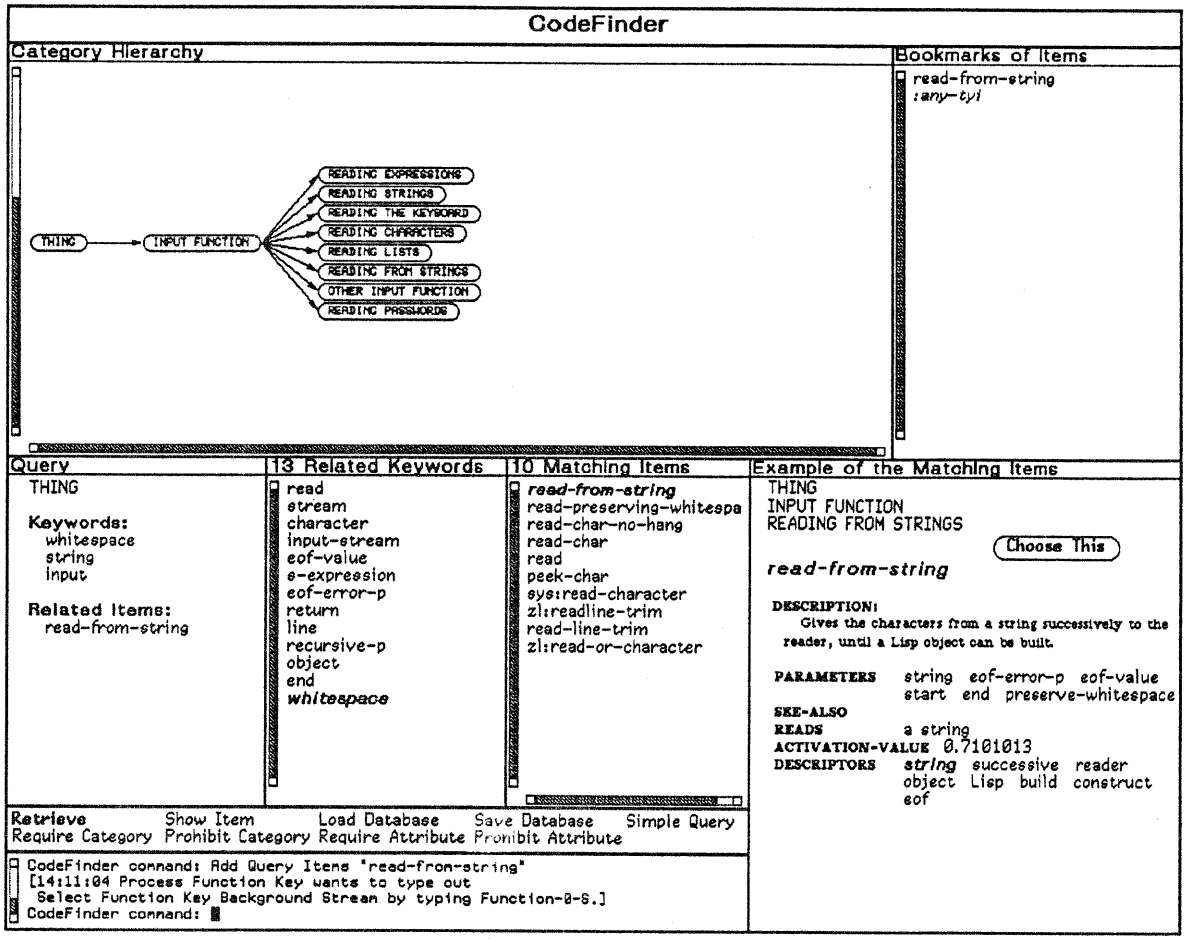

Figure 10: CODEFINDER

CODEFINDER is an information retrieval system in the domain of software objects. In addition to the reformulation techniques of HELGON, CODEFINDER uses a connectionist approach, called spreading activation, to rank the relevance of items retrieved by the system. This gives users a flexible guide to choose which objects should be further examined. The same mechanism is used to rank the relevance of related keywords, which are displayed in the **Related Keywords** pane giving the user additional information for reformulating the previous query. The spreading activation process supports the notion of *soft constraints*. In traditional keyword approaches (including HELGON), if a query does not include keywords associated with a particular object, the object will not be retrieved. In CODEFINDER this problem is reduced through the spreading activation process by retrieving objects that are closely related to the query, but may not contain any of the keywords in the query.

See technical report No. 3.

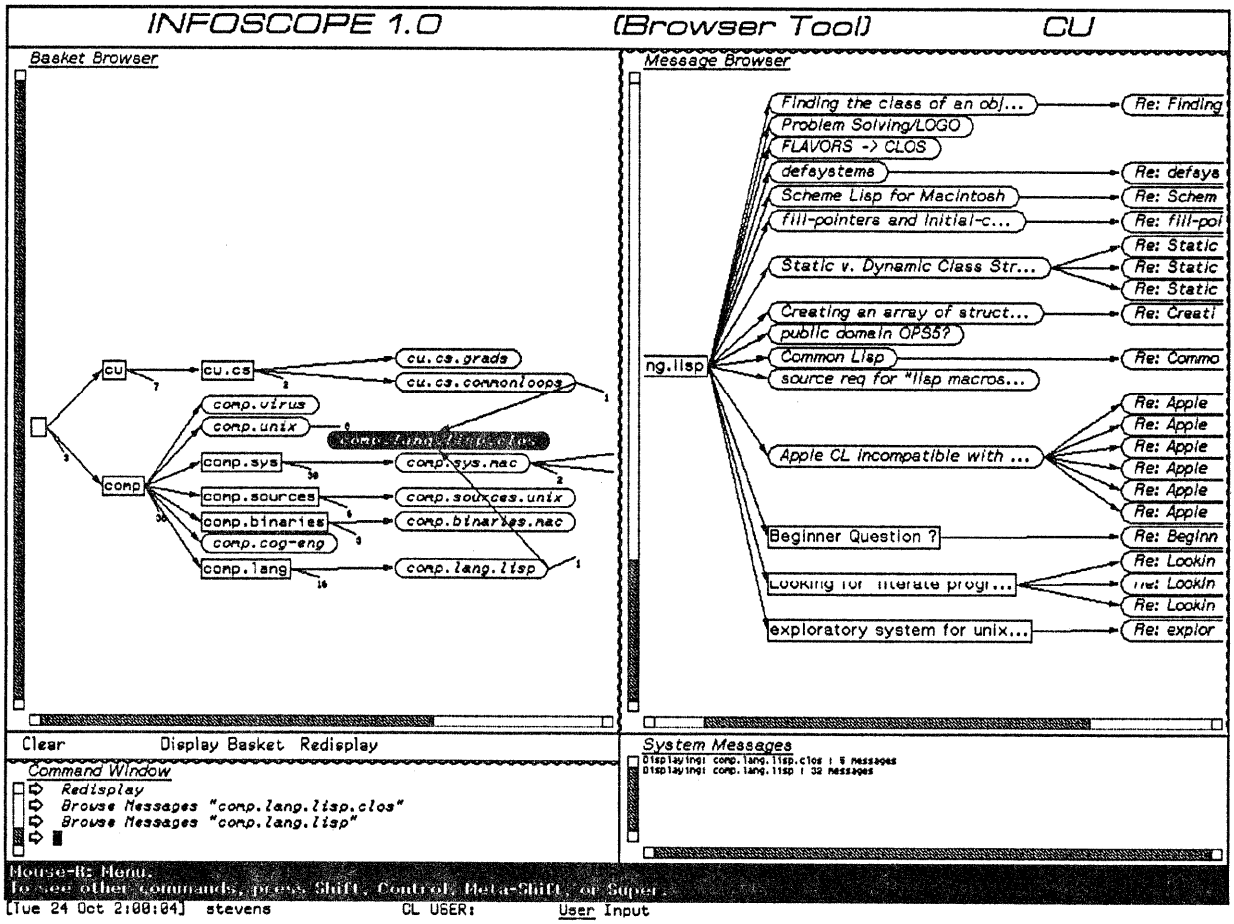

Figure 11: INFOSCOPE

The INFOSCOPE system allows users to define *virtual newsgroups* within the information hierarchy. These virtual newsgroups are created through user-defined filters which ensure that only interesting information is displayed. In this way, users define the beginning of a user model which will be used in future versions of the system in order to offload some of the classification task from the user to an active user modeling component. Virtual newsgroups are created with a single mouse click and the subsequent specification of the keywords which describe interesting conversations. The information space used in this prototype is the USENET news system which we feel is representative of many of the basic information retrieval problems we face.

See technical report No. 5.