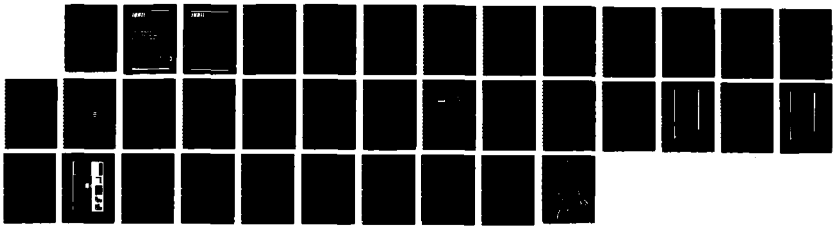
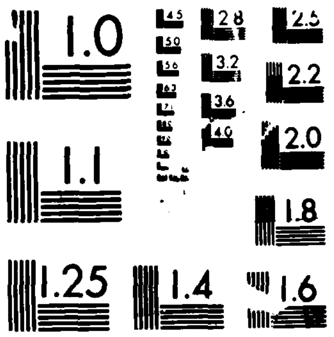
AD-A188 982   A KNOWLEDGE ACQUISITION TOOL FOR THE INTELLIGEN   1/1
PROCESSING OF MATERIALS(U) BDM CORP MCLEAN VA
T A BLAXTON 31 DEC 87 BDM/MCL-87-1207-TR
UNCLASSIFIED   ARO-24902 1-EL-A DAAL03-87-C-0019   F/G 12/5   NL





MICROCOPY RESOLUTION TEST CHART
DS-1963-A

ARO 24902.1-EL-A

(2)

BDM
THE BDM CORPORATION

7915 Jones Branch Drive, McLean, Virginia 22102-3396 ● (703) 848-5000 ● Telex: 901103 BDM MCLN

A Knowledge Acquisition Tool for the Intelligent Processing of Materials

Technical Report

PREPARED FOR U.S. ARMY RESEARCH OFFICE, RESEARCH TRIANGLE PARK, N.C. 27709-2211

DECEMBER 31, 1987

BDM/MCL-87-1207-TR

THE BDM CORPORATION

7915 Jones Branch Drive, McLean, Virginia 22102-3396 ● (703) 848-5000 ● Telex: 901103 BDM MCLN

A KNOWLEDGE ACQUISITION TOOL FOR THE INTELLIGENT PROCESSING OF MATERIALS

December 31, 1987

BDM/MCL-87-1207-TR

Prepared for U.S. Army Research Office, Research Triangle Park, N.C. 27709-2211

UNCLASSIFIED MASTER COPY FOR REPRODUCTION PURPOSES

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) ARO 24902.1-EL-A
6a. NAME OF PERFORMING ORGANIZATION The BDM Corporation 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION U. S. Army Research Office
6c. ADDRESS (City, State, and ZIP Code) 7915 Jones Branch Drive McLean, VA 22102-3396	7b. ADDRESS (City, State, and ZIP Code) P. O. Box 12211 Research Triangle Park, NC 27709-2211
8a. NAME OF FUNDING/SPONSORING ORGANIZATION U. S. Army Research Office 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER DAAL03-87-C-0019
8c. ADDRESS (City, State, and ZIP Code) P. O. Box 12211 Research Triangle Park, NC 27709-2211	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. PROJECT NO. TASK NO. WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
A Knowledge Acquisition Tool for the Intelligent Processing of Materials (Unclassified)

12. PERSONAL AUTHOR(S) Dr. Teresa A. Blaxton

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Interim Technical	FROM ____ TO ____	1987 December 31	31

16. SUPPLEMENTARY NOTATION The view, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Knowledge Engineering, Knowledge Acquisition, Expert Systems, Materials Processing

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Despite recent advances that have been made in expert system building technologies, the process of obtaining knowledge from human experts and coding it into a computing system remains fraught with difficulties. The Knowledge Acquisition Tool (KAT) currently being developed at the BDM Corporation is designed to address this knowledge acquisition bottleneck in the domain of materials processing. KAT offers the knowledge engineer a variety of modes in which to query a materials expert on different aspects of a given materials domain. In the Clarification Mode, the expert specifies the entire process in terms of an AND/OR graph. This structure is converted to a flowchart in the Prediction Mode, where the expert is asked to predict problems that can occur at different points in the process, and to specify limiting conditions. Finally, the Diagnosis Mode provides a dynamic step-through of the process model as defined by the expert, so that s/he can verify the specification of the process and redesign as necessary. (Continued) → [illegible] →

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL | 22b. TELEPHONE (Include Area Code) | 22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Item 19 ABSTRACT (Continued)

Progress made thus far on the KAT project includes a) completion of the overall system design; b) implementation of an initial version of the Clarification Mode; c) partial completion of the Prediction Mode; and d) detailed consideration of KAT's generality to domains other than materials processing.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Despite recent advances that have been made in expert system building technologies, the process of obtaining knowledge from human experts and coding it into a computing system remains fraught with difficulties. The Knowledge Acquisition Tool (KAT) currently being developed at The BDM Corporation is designed to address this knowledge acquisition bottleneck in the domain of materials processing. KAT offers the knowledge engineer a variety of modes in which to query a materials expert on different aspects of a given materials domain. In the Clarification Mode, the expert specifies the entire process in terms of an AND/OR graph. This structure is converted to a flowchart in the Prediction Mode, where the expert is asked to predict problems that can occur at different points in the process and to specify limiting conditions. Finally, the Diagnosis Mode provides a dynamic step-through of the process model as defined by the expert, so that s/he can verify the specification of the process and redesign as necessary. Progress made thus far on the KAT project includes: (1) completion of the overall system design; (2) implementation of an initial version of the Clarification Mode; (3) partial completion of the Prediction Mode; and (4) detailed consideration of KAT's generality to domains other than materials processing.

A KNOWLEDGE ACQUISITION TOOL FOR THE INTELLIGENT PROCESSING OF MATERIALS

A. INTRODUCTION AND OVERVIEW

Today's applications of artificial intelligence are based upon the fact that useful tasks can be accomplished by programs containing large, detailed, specialized knowledge bases. It has become apparent that one of the major obstacles in applying artificial intelligence to specialized domains is the difficulty of creating these knowledge bases, which requires the acquisition of the requisite knowledge from human experts. This knowledge acquisition bottleneck reflects the difficulty of obtaining knowledge from domain experts in a structured fashion and converting it into a form that can be readily accessed by the expert system itself. There have been many recent advances in expert system building technology such as expert system shells, but these advances do not address the knowledge acquisition bottleneck. Building the initial knowledge base remains one of the largest stumbling blocks to successful system completion.

The typical approach to building knowledge bases for AI systems entails prolonged and intensive one-on-one interactions between a programmer and a domain expert. In this setting, the programmer may elicit large amounts of information from the expert, impose his or her own organization onto this information, and encode it into the system. An obvious drawback to this approach is that the programmer learns much more detail about the domain than is likely necessary, and the expert becomes far more knowledgeable about AI programming than s/he might like.

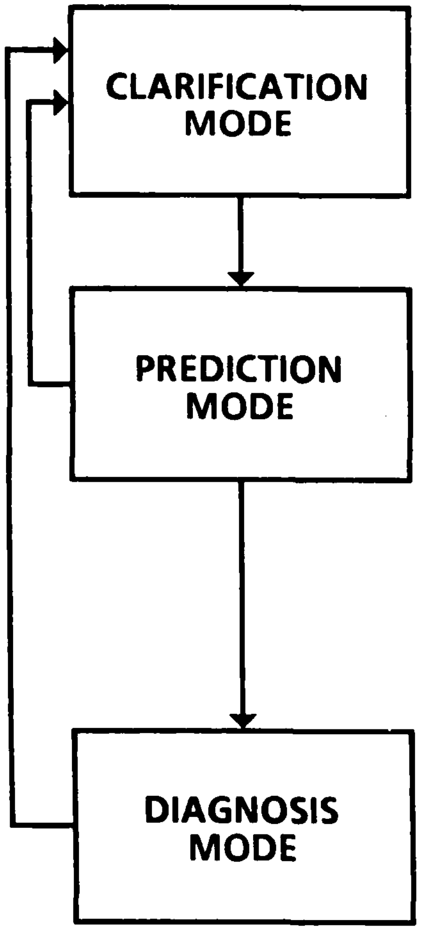
The present Knowledge Acquisition Tool (KAT) project is aimed at investigating remedies for the knowledge engineering problem. KAT is being designed for use by a programmer as a tool to aid in the knowledge acquisition process, providing structured mechanisms for eliciting information about materials processing. It is important to note from the outset that KAT will not be used as a tool for building an expert system in materials processing. Rather, KAT will be used by the programmer to build up a knowledge base that can then be handed over to designers who will be building an expert system. Furthermore, it is important to realize that although KAT will automate the knowledge engineering process to some degree, it should not be considered a replacement for the knowledge engineer. It is best thought of as a tool that the knowledge engineer uses to query the expert in a structured manner. Such a tool will constitute a marked advance in the knowledge engineering process.

Though KAT is not an expert system, it is a sophisticated tool embodying the best methodologies known for process specification, both in top-down and bottom-up fashion. The top-down portion is based on the highly successful Higher Order Software approach (Hamilton & Zeldin, 1980; Harel, 1980) used in complex software development for NASA, extended slightly to allow parallel processes (Reeker, 1986). The bottom-up portion features flowcharting that can be rearranged dynamically and transformed back to the top-down AND/OR representation.

KAT can be used to extract knowledge both through explicit and implicit queries of the expert, who will be aided by the knowledge engineer. Research in the area of human memory has suggested that procedural knowledge might best be extracted by implicit means (e.g., Graf & Schacter, 1985; Roediger & Blaxton, 1987). In specifying processes, however, all knowledge is not procedural, as a number of declarative facts will be germane to the process. (For a discussion of the distinction between procedural and declarative knowledge, see Kolers & Roediger, 1984.) For this reason, KAT has been designed to provide facilities for obtaining both types of knowledge in a natural manner.

As shown in Figure 1, the expert will interact with KAT in three different "modes" called Clarification, Prediction, and Diagnosis. These modes will be used iteratively to provide a structure for knowledge acquisition. The Clarification Mode (Figure 2) is a framework that allows the expert to tell the system what his or her mental model of the process is in terms of subprocesses. This is the top-down, or process decomposition portion of the knowledge acquisition activity. As already mentioned, the

THE OVERALL STRUCTURE OF KAT



- DESCRIPTION OF PROCESS DECOMPOSITION IS CREATED IN THE FORM OF AN AND/OR GRAPH
- LOGICAL RELATIONSHIPS AMONG PROCESSES ARE REPRESENTED EXPLICITLY

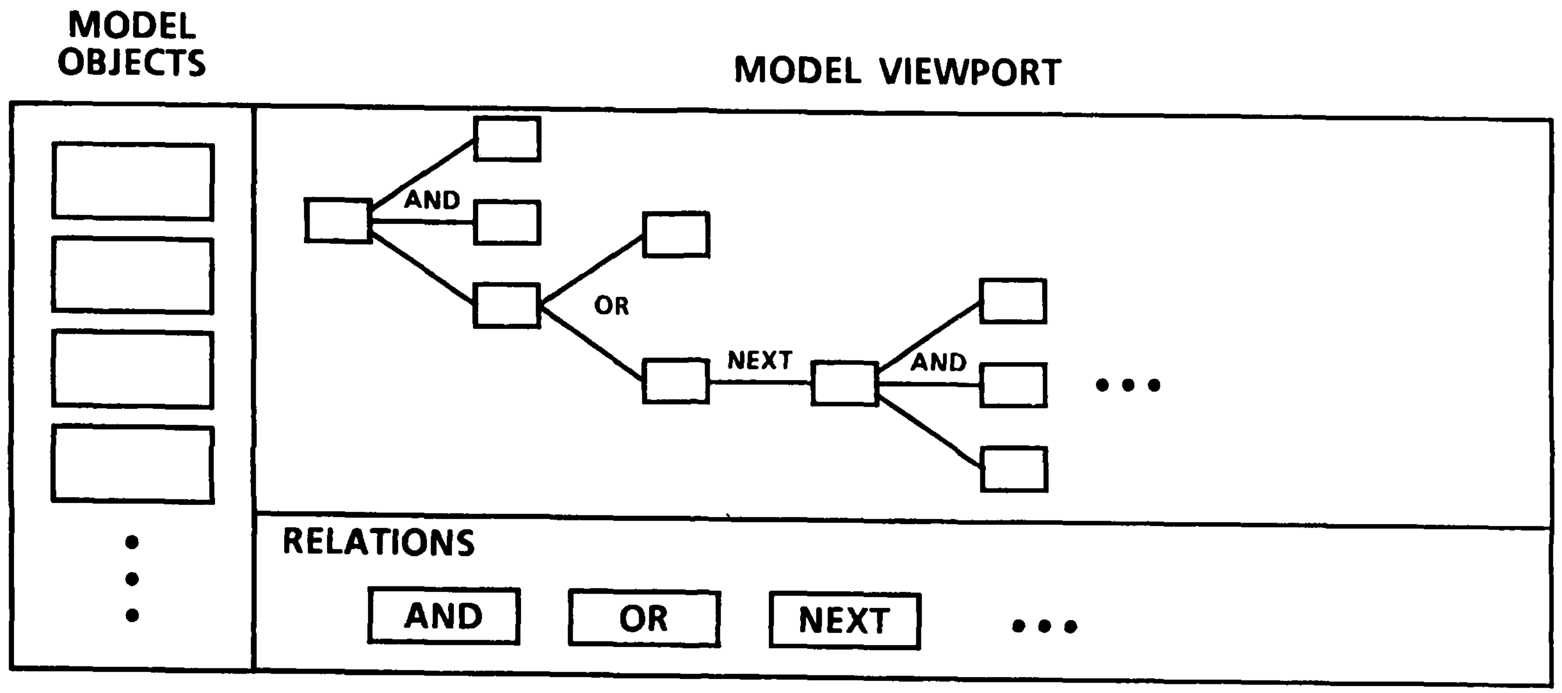
- INFORMATION IS PRESENTED IN FLOWCHART FORMAT, AND EXPERT CHECKS
  - INPUTS & OUTPUTS
  - RELEVANT ATTRIBUTES
  - PROCESS RULES
- INFORMATION FORMAT IS CHANGED TO SHOW TEMPORAL ORDERING; PROCEDURAL KNOWLEDGE ELICITED

- DYNAMICALLY STEP THROUGH KNOWLEDGE STRUCTURE, MAKING CHANGES AS NECESSARY
- IMPLICIT INTERACTIONS WITHIN THE PROCESS MODEL ARE EXPLICATED

*7-1-MCL3-000501-01

Figure 1. The Overall Structure of KAT

CLARIFICATION MODE DESIGN



- EXPERT STARTS FROM SCRATCH AND DECOMPOSES PROCESS INTO A LOGICAL PROGRESSION OF SUBPROCESSES
- ALTERNATIVE SUBPROCESSES ARE INDICATED WHERE APPROPRIATE (*OR*s)
- RESULT IS AN *AND/OR* GRAPH OF THE ENTIRE PROCESS

*7-1-MCL3-000501-02

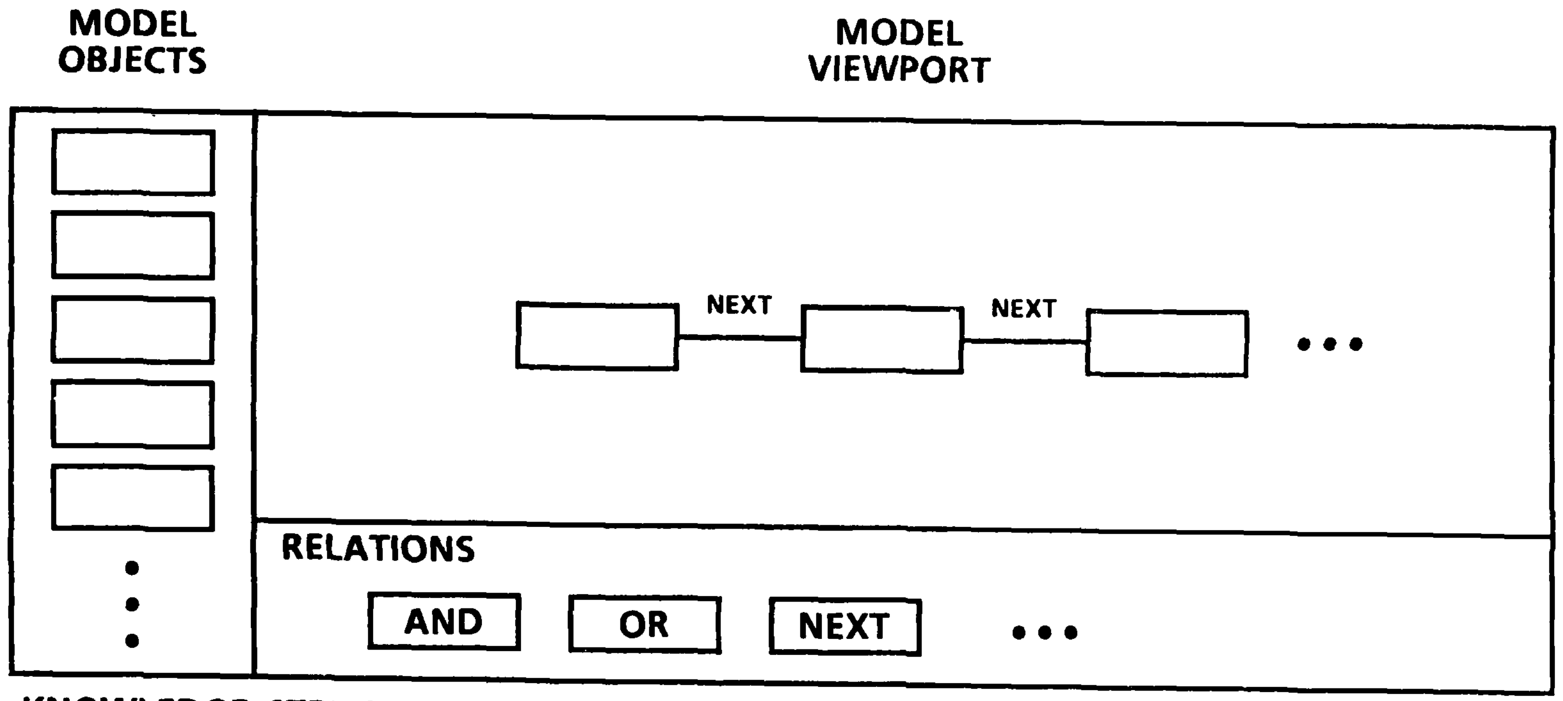
Figure 2. Clarification Mode Design

result is an AND/OR graph. Where OR nodes are used, the branches will be labeled by conditions that will cause one alternative to be chose.

The Prediction Mode (Figure 3) provides another view of the model built during the Clarification Mode. The knowledge structure created earlier will be presented in such a way as to emphasize the temporal flow of the process, which will expose "holes" that were previously missed. Time and sequencing relationships that may only be implicit in the AND/OR representation are now shown explicitly, though the logical subsetting apparent in the AND/OR graph is suppressed. (Although the knowledge structure depicted in Figure 3 shows no AND/OR relations, this will not always be the case.) During this mode the user will be asked questions concerning particular conditions that must be satisfied in order for the process to continue. Various facilities for encoding this information are being developed and will be discussed later.

Once the expert has iterated between the Clarification and Prediction Modes several times, a fairly well-specified knowledge structure will be in place and the Diagnosis Mode will be invoked. The Diagnosis Mode (Figure 4) will provide a means of "stepping through" the process model in a dynamic manner. Where the model is not yet adequately specified, this mode will expose the need for more detail. In a sufficiently knowledge-rich environment, the Diagnosis Mode could actually simulate the process taking place, although it has not been designed to be a simulation environment in the normal sense. For a materials processing application, the amount of knowledge required for a simulation would be considerable and would include the sort of qualitative physical knowledge studied by De Kleer (1984) and Kuipers (1984). Interest exists in this aspect of KAT for the future, but the present Diagnosis Mode is designed to do an event-stepped "tour" of the process specified. In this "tour", the expert will have to verify that the model is indeed reflective of the true physical world.

PREDICTION MODE DESIGN

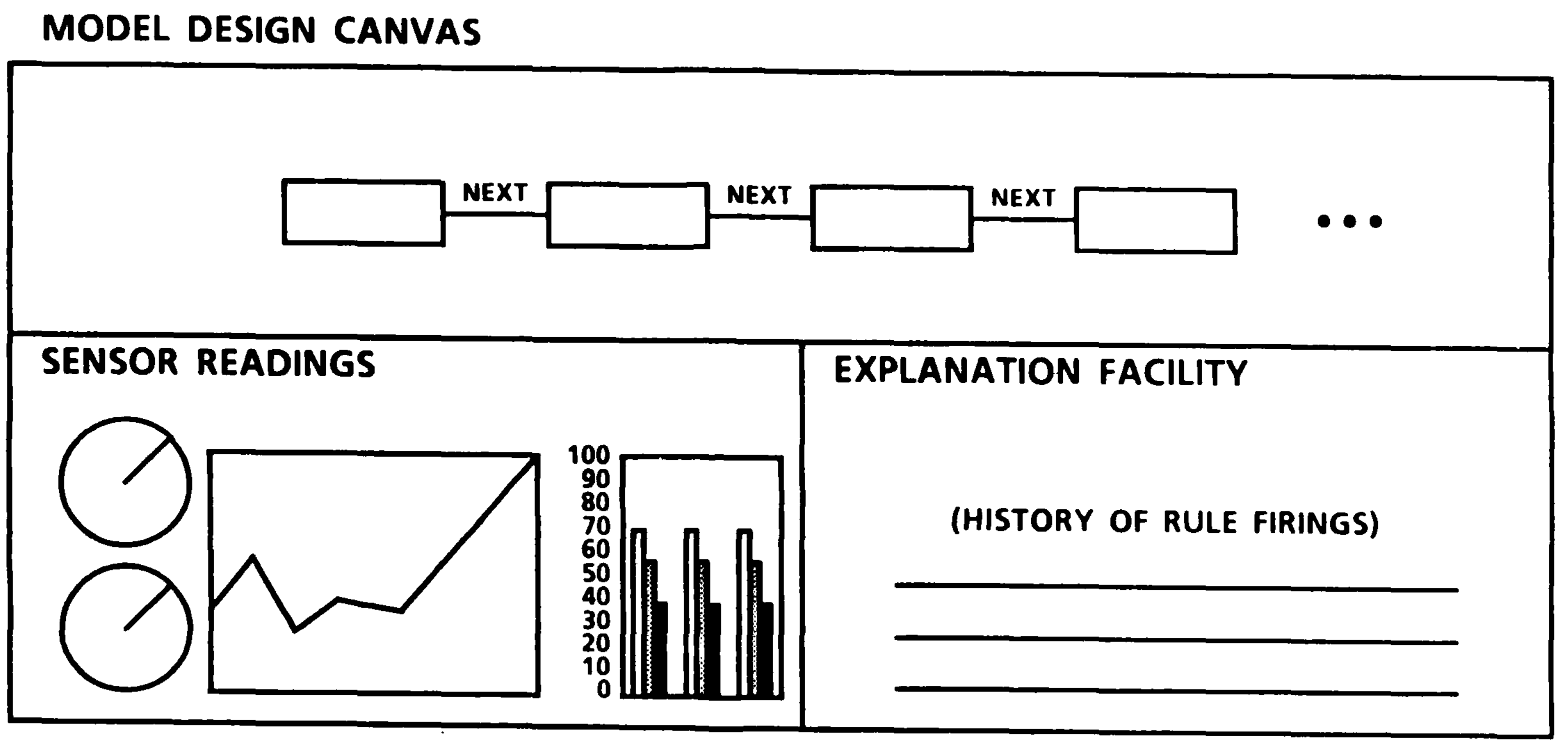


- KNOWLEDGE STRUCTURE CREATED IN THE CLARIFICATION MODE IS REDISPLAYED IN A TEMPORAL (FLOWCHART) FORM
- FOR EACH SUBPROCESS, THE EXPERT IDENTIFIES:
  - INPUTS AND OUTPUTS
  - OTHER RELEVANT DATA STRUCTURES
  - RULES WHICH DEFINE RELATIONSHIPS AMONG SUBPROCESSES, ETC.
- TEMPORAL ORDERING MAY BE EASILY MODIFIED

*7-1-MCL3-000501-03

Figure 3. Prediction Mode Design

DIAGNOSIS MODE DESIGN



- KNOWLEDGE STRUCTURE IS PRESENTED IN FLOWCHART FORM
- EXPERT SEES A DYNAMIC STEP-THROUGH DISPLAY OF THE PROCESS WITH RELEVANT SENSOR READINGS AND THE HISTORY OF RULE APPLICATIONS
- THIS STEP-THROUGH SERVES AS AN IMPLICIT QUERY, PROMPTING THE EXPERT TO REDESIGN UPON NOTICING INCOMPLETE OR FAULTY SPECIFICATIONS

*7-1-MCL3-000501-04

Figure 4. Diagnosis Mode Design

3. KAT VERSUS TRADITIONAL KNOWLEDGE ENGINEERING

The traditional approach to knowledge engineering usually involves an extended interviewing process whereby the knowledge engineer asks the expert as many questions as possible about the domain of interest. As shown in Figure 5, these questions are limited in their utility in several ways. First, the queries are explicitly formed and are presented to the expert in such a manner as to presuppose a particular type of response. Second, the fact that the queries can be posed at all by the knowledge engineer implies that s/he has some understanding of the domain and its important information elements. The knowledge engineer's acquisition of this domain knowledge is not a trivial task. (After all, if the domain were easily understood, there would be no need to consult an expert.) In all likelihood, a large amount of time will be spent in coming up to the level of understanding that enables meaningful interactions with the expert.

The third and most serious drawback to the questions that the knowledge engineer poses to the expert is that they reflect the knowledge engineer's own mental model of the domain and not the expert's. The knowledge engineer deems particular questions to be important because they fill in missing information in his or her personal conception of the problem domain. Since the knowledge engineer is not experienced in this domain, it is likely that s/he will focus on concepts that are only tangential to the main issues, and will bias the construction of the knowledge base. Since the expert is not experienced in AI programming, s/he will not be able to look at the knowledge structure as it is developed and make corrections as to the proper hierarchical arrangement of knowledge base elements. One might imagine how a few mistakes of this type early in the knowledge acquisition process could lead to deeply entrenched biases in the knowledge structure that would be difficult to compensate for at a later time.

In contrast to this approach, KAT offers several modes in which the expert may be queried about the domain. Some of these queries will be explicit whereas others will be implicit. For example, in the Prediction

COMPARISON OF KAT WITH TRADITIONAL APPROACHES TO KNOWLEDGE ENGINEERING

TRADITIONAL KNOWLEDGE ENGINEERING	KAT
LIMITED (EXPLICIT) TECHNIQUES FOR ELICITING KNOWLEDGE	LARGE NUMBER OF TECHNIQUES FOR ELICITING KNOWLEDGE (IMPLICIT AND EXPLICIT)
INFORMATION RECODING: - TIME CONSUMING - LOSS OF INFORMATION WHEN SWITCHING FORMATS - KNOWLEDGE MAY END UP IN AN INAPPROPRIATE FORM	- NO RECODING OF INFORMATION; ALL INPUTS ARE PUT DIRECTLY INTO THE KB - SYSTEM CONTROLS THE TYPE OF DATA STRUCTURE USED TO REPRESENT INFORMATION
INTEGRATION OF INPUTS FROM MULTIPLE EXPERTS IS DIFFICULT	SPECIALLY DESIGNED FOR USE WITH MULTIPLE DOMAIN EXPERTS
PROCESS FOR VALIDATING THE KB IS ILL-DEFINED	DIAGNOSIS MODE OF QUERY IS DESIGNED TO ASSIST IN THE VERIFICATION PROCESS

Figure 5. Comparison of KAT With Traditional Approaches to Knowledge Engineering

Mode the expert will be asked explicitly to identify sensors that are of interest at a particular point in the process. In the Diagnosis Mode, however, readings from these sensors will be used to make inferences concerning whether the process is properly specified. This second type of interaction involves what may be termed an implicit query in that the expert is watching the process in action and redesigning as necessary, rather than responding to overt questions from the knowledge engineer.

A second class of problems that one encounters when using traditional approaches to knowledge engineering involves the recoding of the information provided in the expert's responses to a form that the AI system can use. In most cases, the expert is interviewed at length at some place other than the software development site. Following an interview session, the knowledge engineer has to work through and organize the information gleaned during the interview and decide how best to encode it into the already existing knowledge structure. Since the expert is not present when this recoding takes place, the knowledge engineer runs the risk of either omitting important information, coding the information into an inappropriate form, or both.

KAT is designed to avoid these recoding problems. Specifically, since the knowledge engineer will be using KAT directly during the knowledge acquisition process, the information gathered from the expert may be directly encoded into the knowledge base as it is elicited. Furthermore, the queries posed by the system will be designed in such a way that their answers will naturally fit into a particular data structure format. That is, the system will already be expecting that a response to a particular question will take the form of a rule (or a frame, as the case may be). Facilities for entering the information into the system will be available to automate this process as much as possible. Thus, errors resulting from miscoding may be more easily avoided. Finally, since the expert will be present when the information is entered into the knowledge base, s/he will have the opportunity to suggest corrections as appropriate.

A classic problem encountered during knowledge engineering efforts is the acquisition of information from multiple experts. In many cases, the experts fail to agree on important points and the knowledge engineer is not in a position to judge the opinions of one against the other. A more serious problem lies in the fact that the mental model of the domain varies considerably from expert to expert. This is reflected in differences in emphasis that are imparted to the knowledge engineer during the acquisition process. The fact that one concept is considered more important than another can have a large effect on the knowledge base structure. Consequently, knowledge bases engineered from multiple experts can "look" different from one another, even though there may be few outright contradictions.

KAT is being designed to provide assistance with this problem to the knowledge engineer. By providing facilities for experts to specify and edit models of the materials process, a handy tool for observing differences in emphasis imparted by different experts will be available. There will be opportunities for an expert to compare directly his or her formulation of the domain with that specified by other experts. Once exposed, these disagreements can be resolved by revising one or more of the knowledge structures, or at least can be marked for future reference.

One final class of problems that KAT addresses is that of verifying the knowledge structure once it is in place. In normal practice, it is sometimes quite difficult to check that the information encoded in the system actually conveys the meaning intended by the expert. One way in which this problem will be addressed in KAT is in the Diagnosis Mode. During Diagnosis, the expert will see an automated step-through of the process as it has been specified up to that point. If all is well, the process should flow according to the expert's expectations. Deviations from those expectations, however, will indicate either that information has been coded in an improper way or that information is missing altogether. Once these problems are detected, the expert can immediately direct their correction on-line. S/he can then check to see that proposed solutions have indeed eliminated the problem by re-running the step-through.

C. KAT'S FIRST APPLICATION DOMAIN: CHEMICAL VAPOR INFILTRATION

Although the design of KAT is general enough to be implemented in any materials processing domain, the first demonstration will be built in the area of Chemical Vapor Infiltration (CVI). An extensive literature search has been conducted to determine the status of current CVI research methodologies. After reviewing the pertinent literature, attention was focussed primarily on the work of the Oak Ridge National Laboratory (Caputo & Lackey, 1984; Caputo, Lackey, & Stinson, 1985).

The objective of the Oak Ridge work has been to produce composites by first making a ceramic fiber preform and then synthesizing the ceramic matrix by infiltrating this fibrous preform using a low stress, low temperature process that would not damage the high strength ceramic fibers. Previous CVI work indicated that required processing times have been extremely long, sometimes lasting several weeks. A goal of this work, then, has been to reduce processing times to a matter of hours so that composite costs would be lowered sufficiently to make such materials cost-effective for engineering applications.

As shown in Figure 6, the CVI method used at Oak Ridge involves a new approach that combines the use of the thermal-gradient and pressure gradient processes. A critical portion of their equipment includes a gas-injector and preform holder. A graphite preform holder is placed on top of a water-cooled metal holder. The graphite holder is cooled sufficiently by contacting the water-cooled metal holder to prevent deposition on the bottom and side surfaces. Reactant gases flow up through the water-cooled gas injector, through the preform, and when the reactants are sufficiently heated (near the top of the preform) the chemical vapor deposition reaction occurs. Initially, the gases are able to flow both axially and radially through the preform. However, when the top surface becomes coated, the gases flowing up into the preform must then flow radially through the preform to the annular void space around the preform, and then escape through the holes in the retaining ring. As a result, the fibers are coated uniformly in the preform, thus forming the matrix phase of the composite. As

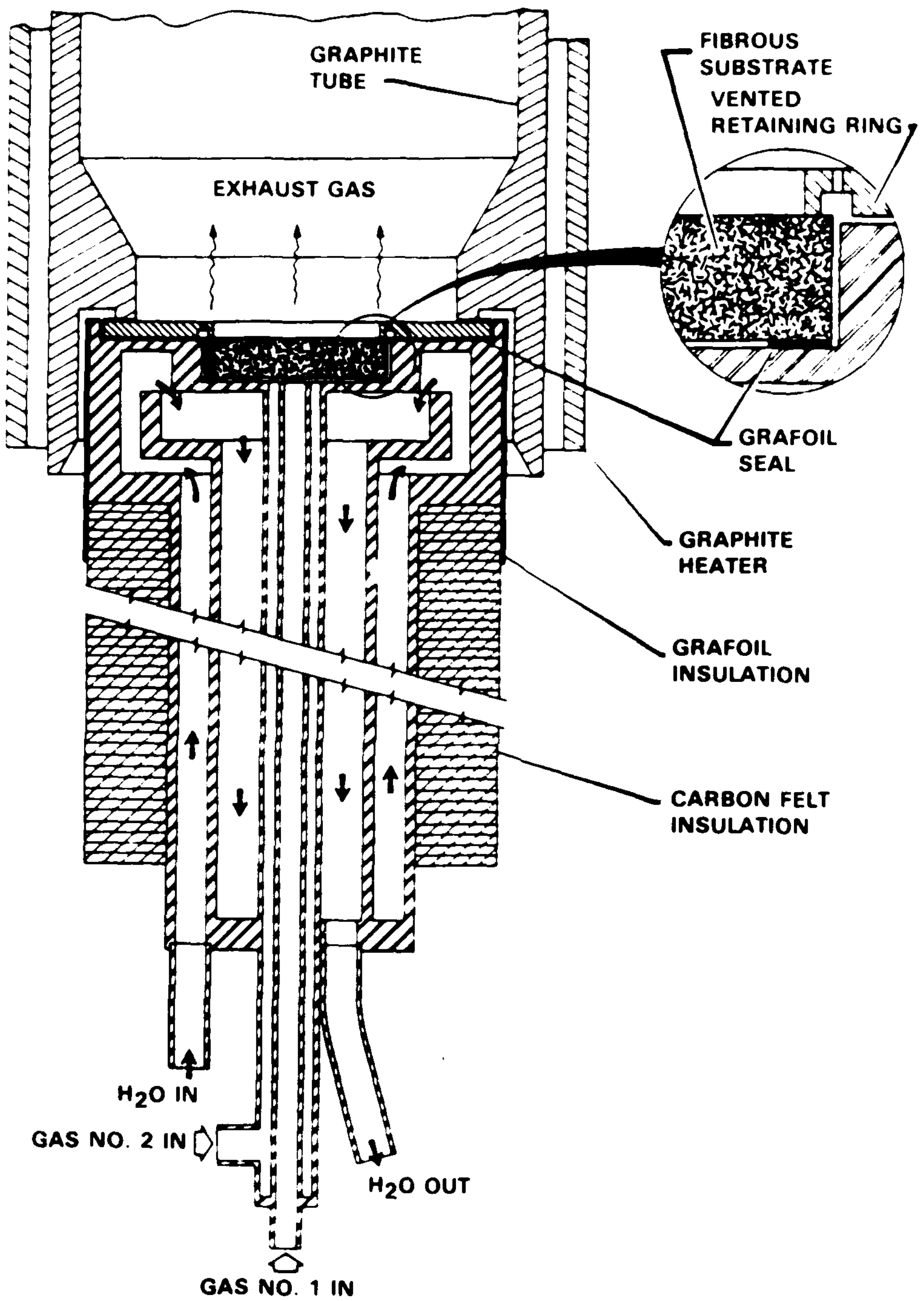


Figure 6. Example of a Chamber Used for CVI Using Thermal Gradient and Gas Flow-Through System (From Caputo and Lackey, 1984, Ceramic Engineering Sciences Proceedings)

a matrix deposition continues, the thermal conductivity of the infiltrated portions of the preform increases and the deposition front moves progressively from the top of the preform toward the bottom and circumference.

This initial characterization of the CVI process has been taken from published articles and reports. Important information not included in such documents includes such things as: type and location of sensors in the reactor environment; heuristic knowledge about interpretation of these sensor readings; and diagnosis of final products to determine process redesign. In order to gain some insight into these areas, BDM personnel have visited Caputo's lab at Oak Ridge and have attended conferences on CVI. Information obtained from these and other interactions will be used to help design the initial implementation of the KAT system.

D. INITIAL IMPLEMENTATION OF KAT

KAT is being built at BDM on a Symbolics 3670 machine in the KEE and SIMKIT environments. KEE is an expert system building shell sold by Intellicorp. It provides many facilities to aid AI programmers in the generation of frame-like objects and rules. In addition, it has graphics and specialized mouse functions which are useful in constructing user interfaces. The SIMKIT system, which resides on top of KEE, provides a host of facilities for writing event-driven simulations, including a clock, event calendar, data collectors, and various mathematical tools for data analysis. The event-driven nature of the SIMKIT simulation environment makes it ideal for the materials processing domain.

The following is a brief description of the features being implemented in KAT using both functions available from the KEE and SIMKIT environments as well as other code generated at BDM. Some of these features have already been implemented, and others are currently being designed. Specifically, BDM has completed versions of the CVI Library and the Clarification Mode. The Prediction Mode is partially completed at this point. The next implementation efforts will be directed toward completing the Prediction Mode and implementing a first version of the Diagnosis Mode.

1. The CVI Library

Before one can build a process model in SIMKIT, a library must first be constructed which contains all objects and rule classes that will be used in the final system. The review of the CVI literature conducted by BDM has provided a preliminary set of objects with which a library has been constructed. As shown in Figure 7, such objects include reactor types, preform types, information about thermal and pressure gradients, etc. For each of these objects, further information may be stored in internal "slots". For instance, the object corresponding to a one-dimensional fibrous preform might have slots for such features as size, material takeup, initial density, optimum density, and reactive gas, to name a few. Each of these slots is allowed to take on certain values. These values can be set and reset for any particular materials design scenario.

The aim in constructing this initial library has not been to represent exhaustively every element of interest in the CVI process. That task will be left to the domain expert when KAT is actually used to build the CVI knowledge base. Rather, the intent is to anticipate, to a certain degree, some general concepts that are needed to describe the CVI process. These basic components will then be in place in the system when the knowledge engineer and the expert sit down for the first time to create a CVI model. Building this initial library will not only save the expert time, but will provide some guidance for the expert's early efforts in partitioning the CVI world into objects.

2. The Clarification Mode

The expert's task in the Clarification Mode is to create an AND/OR graph of the CVI process. At the outset of this mode, certain objects corresponding to units in the CVI library are available in an object window (see Figure 8). If the expert wishes, new objects not already in the library can be created at any time and placed in this object window. The expert indicates that a particular object should be moved into the main viewport and included in the AND/OR graph. When the corresponding icon is moused in the object window, a menu will appear displaying the

CVI LIBRARY

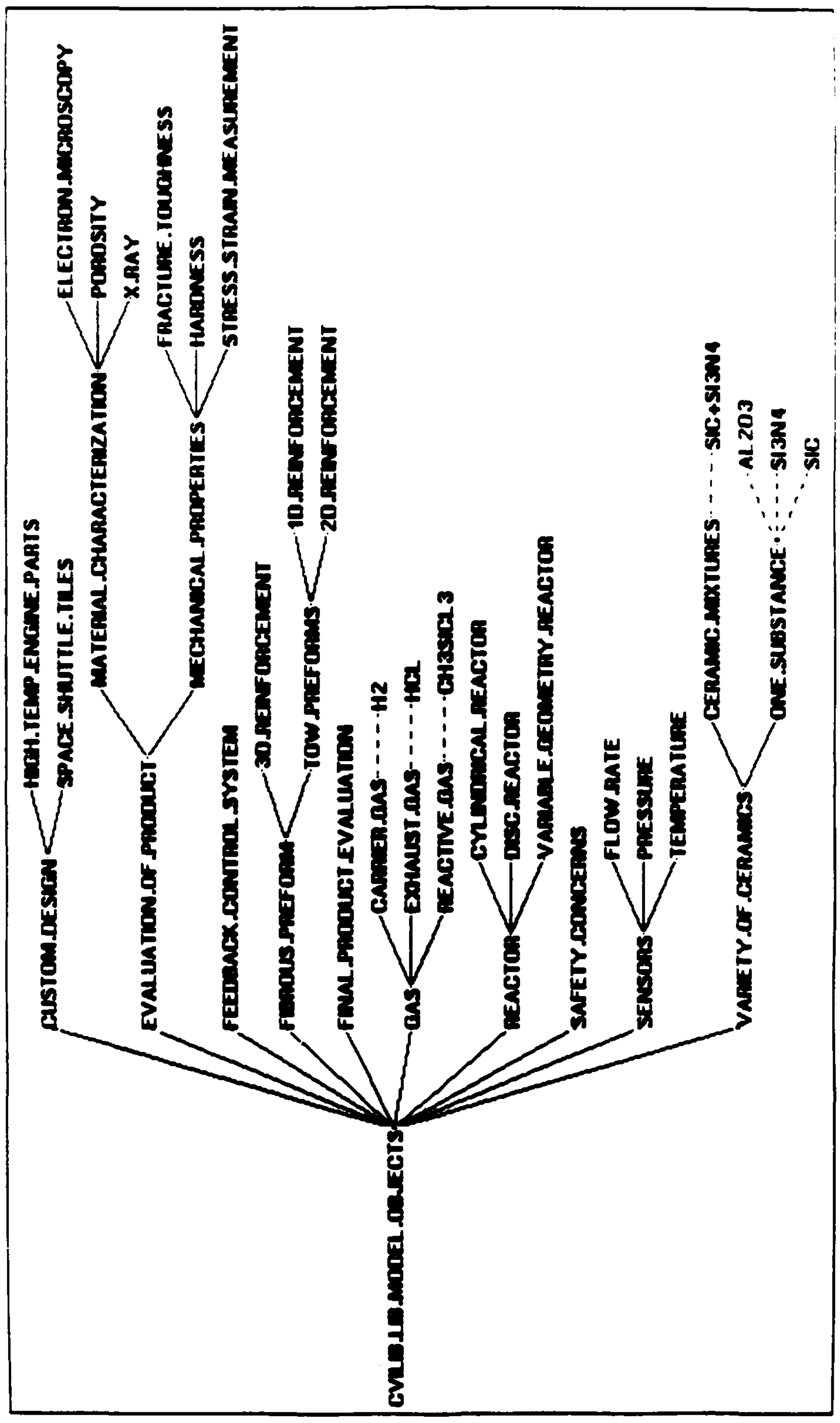


Figure 7. CVI Library

KAT INTERFACE FOR CLARIFICATION MODE

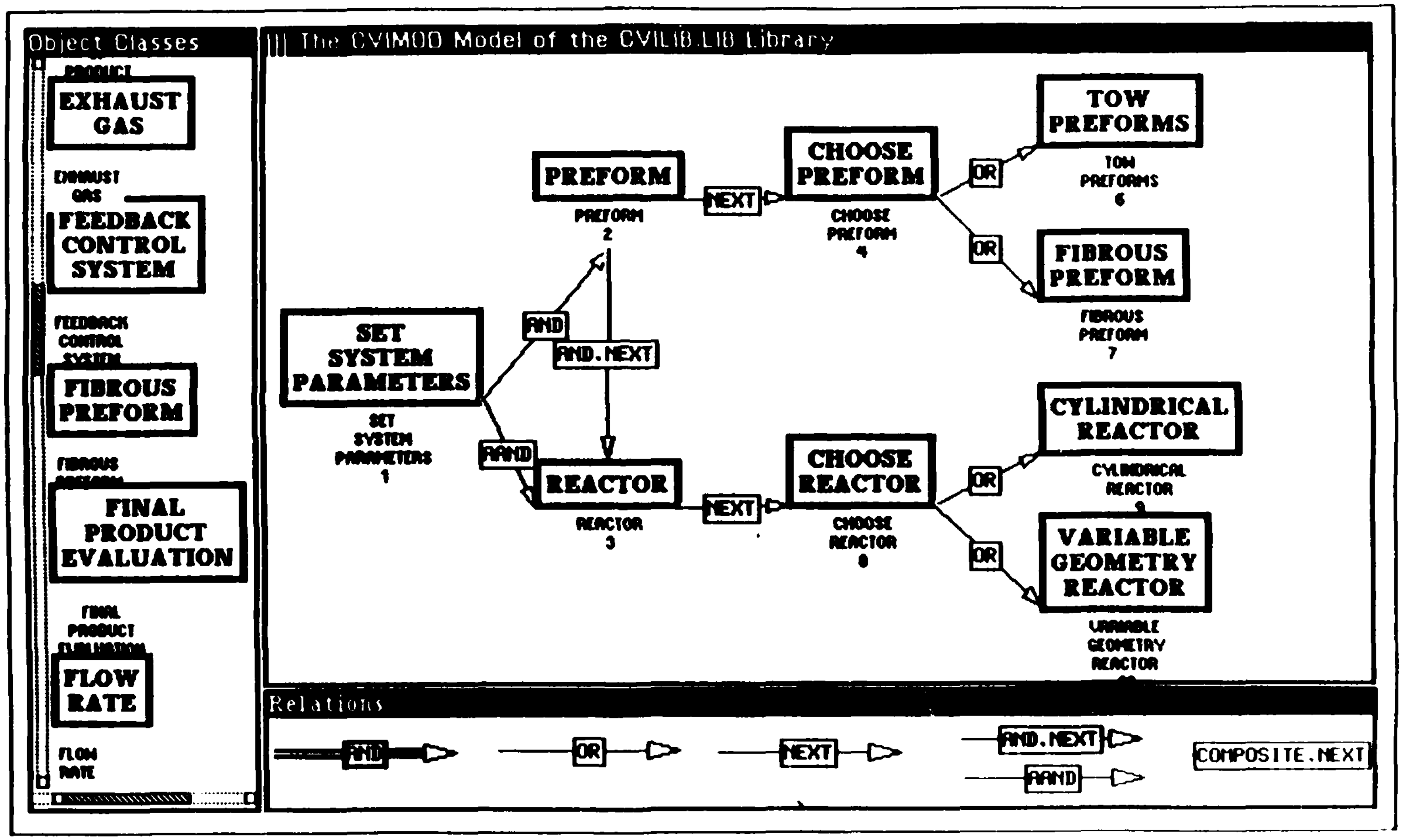


Figure 8. KAT Interface for Clarification Mode

slots and values of that object before its placement in the viewport is confirmed.

Once several objects are in place in the viewport, the expert can indicate the type of relation that should exist between them. These relations include AND, OR, and NEXT, and are displayed in the relations window below the viewport. (Note that other relations are displayed as well. While not used during Clarification, these are of interest in the Prediction Mode.) The user mouses on the desired relation, and then mouses on the objects to be linked with that relation. The system will then prompt for other objects to be included in the relation, and the final configuration will be displayed with arrows indicating relations that have been defined between objects. This graph creation will continue until the expert has thoroughly specified the CVI process.

3. <u>The Prediction Mode</u>

The information entered into the system during the Clarification Mode is converted into a flowchart representation before being shown again in the Prediction Mode (see Figure 9). Code has been generated which collapses AND and OR tree structures down into single composite objects. These composite objects are displayed with simple "next" relations in the flowchart format, thus emphasizing the temporal flow of the process.

For each subprocess point in the flowchart, the expert will be asked which sensors are relevant and what their normal ranges should be. In addition, the expert is asked questions concerning his or her interpretation of extreme sensor readings at each subprocess point. The expert is asked what very high or very low readings would indicate at this point in the process. The responses to such queries will take the form of rules. For example, the expert might say, "If the temperature of the reactive chamber is greater than 1200 degrees, then the infiltration of the preform will be uneven." This identification of potential problems in the Prediction Mode will likely prompt the expert to redesign the process in some cases. Thus, facilities for rearranging, adding, deleting, and creating new objects are also available during the Prediction Mode.

KAT INTERFACE FOR PREDICTION MODE

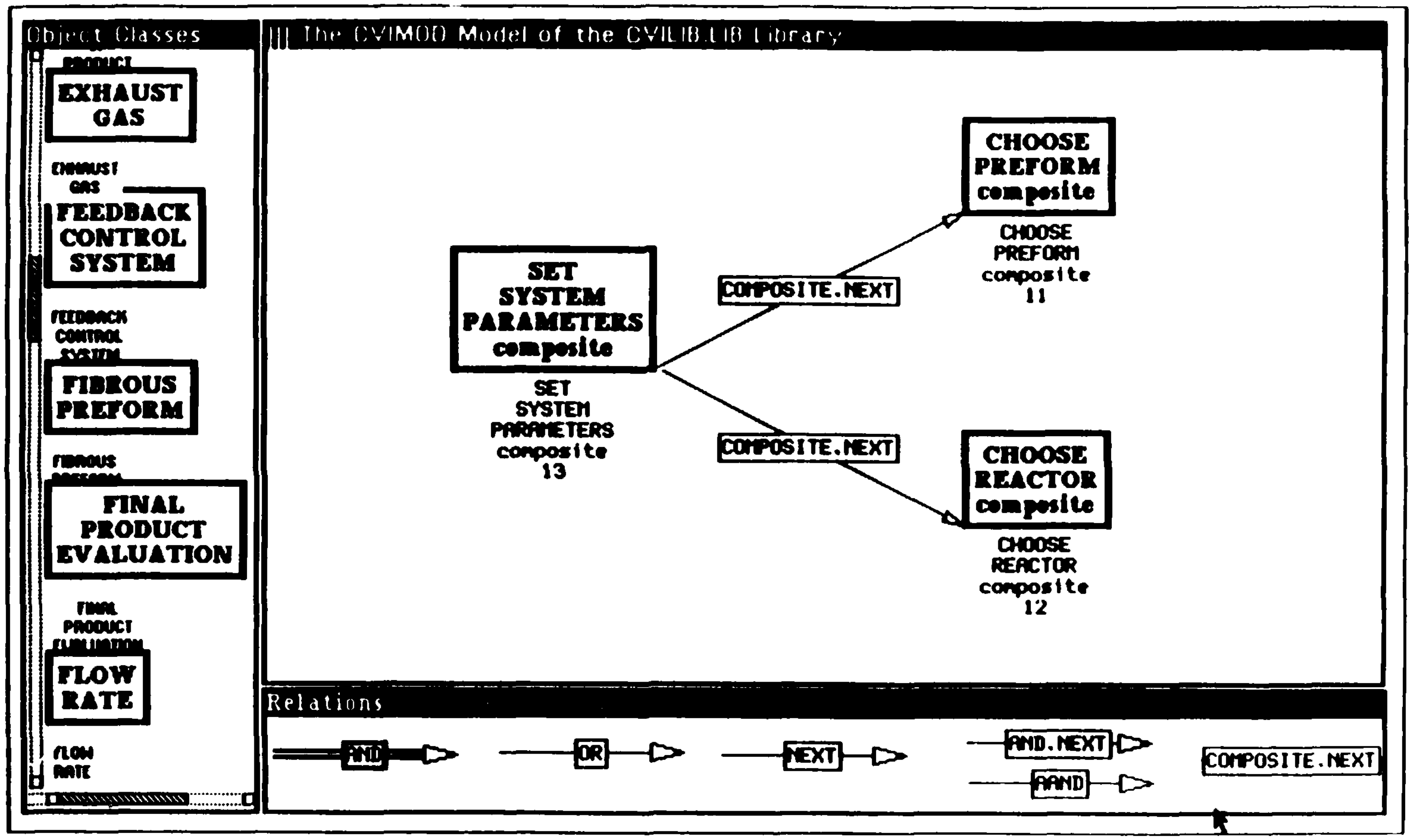


Figure 9. KAT Interface for Prediction Mode

The task of handling free-form user inputs and converting them into acceptable rule form is a complex one. There are many problems to consider, such as how to have multiple antecedents and consequents, how to handle variables in rules, how to choose from among the several rule types available in the KEE system, and how to incorporate newly created rules into the already-existing knowledge structure. To address these problems, a menu-driven facility is being designed that will handle the construction of rules in a partially automated manner. In its early version, the rule builder will by used by the knowledge engineer to construct standard If-Then rules that conform to the syntax dictated by KEE. Later, facilities will be added for constructing LISP expressions and methods.

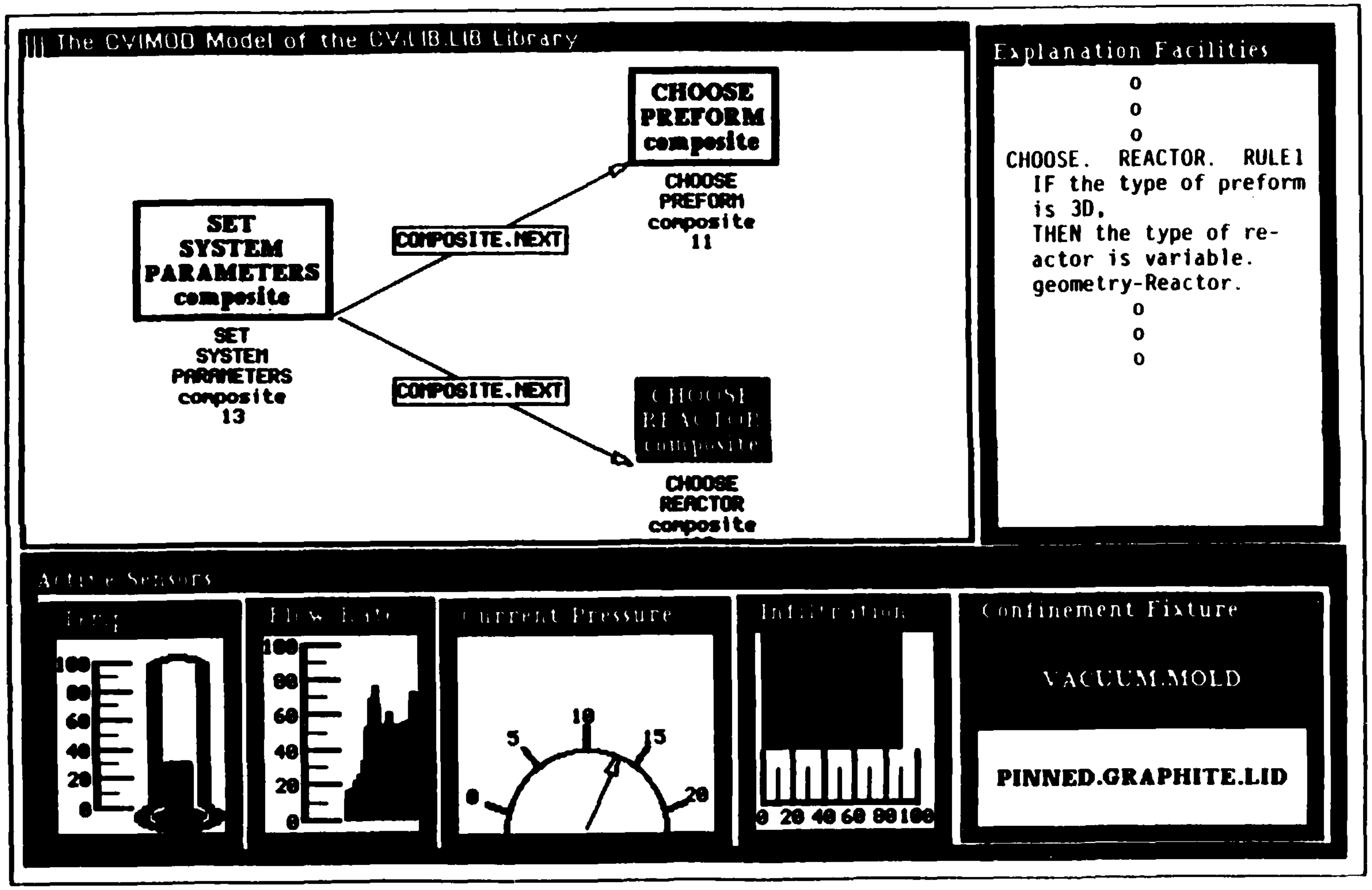
Once the expert has been queried about each subprocess in the flowchart, the knowledge engineer will return to the Clarification Mode. In the Clarification Mode, the knowledge structure will again be presented in an AND/OR format and the expert will further specify the process. It is expected that switching between the AND/OR graph and the flowchart displays will prompt the expert to think of details to add to the model created up to that point. These details might not be remembered if the model was always presented in one format.

4. <u>The Diagnosis Mode</u>

The system features described thus far have already been implemented or, in the case of the rule builder, are currently being implemented. In this section a design for the Diagnosis Mode is presented, in addition to the technical issues to be addressed in its development.

Once the knowledge engineer has iterated through the Clarification and Prediction Modes several times with the expert, a good deal of information about how the process occurs will be represented in the knowledge structure. In the Diagnosis Mode, the expert will have a chance to look at a step-through simulation of the process. Specialized graphics will have been created earlier in the Prediction Mode for displaying sensor readouts at this time (see Figure 10). In addition, the expert will be able to observe directly the consequences of any process rules defined during the Prediction Mode. If at any time the expert perceives the

KAT INTERFACE FOR DIAGNOSIS MODE



```
      o
      o
      o
CHOOSE.  REACTOR.  RULE1
  IF the type of preform
  is 3D,
  THEN the type of re-
  actor is variable.
  geometry-Reactor.
      o
      o
      o
```

Figure 10. KAT Interface for Diagnosis Mode

process to be going awry, s/he can stop it immediately and redesign as necessary. Such redesign could include any number of measures such as adding, deleting, or rearranging sensors, rules, or objects.

Note that the form of query in the Diagnosis Mode is more implicit than in the previous modes. There is no program of queries to which particular types of responses are expected. The expert is not being given direct questions about the structure of the process. Rather, s/he is allowed to observe a step-through simulation as defined by information provided thus far, and make determinations about how this design could be improved. Note also that the term "simulation" is not being used in the traditional sense. Rather, it refers to a facility in which a progression of previously defined states is displayed through, among other things, changes in sensor icons. In order to have a true simulation of the CVI process, one would first need a structural model on which to build. Such a model may indeed be the final output of KAT, but would not be specified by the knowledge structure at this point.

E. APPLYING KNOWLEDGE ENGINEERING TO SOFTWARE ENGINEERING

Having completed the initial design and some of the initial implementation of KAT in a materials processing domain, BDM is exploring the possibilities of KAT's application in other unrelated areas. In the most general sense, KAT should be a useful tool in any domain embodying process knowledge. These domains vary widely and might include materials selection, factory automation, planning, etc. One potential application area that appears to hold a great deal of promise is that of software engineering. In the remaining sections of this report, software engineering is defined and the ways in which it could benefit from a tool such as KAT are discussed.

1. Process Knowledge and Programming

The enterprise of software engineering is dedicated to creating programming methodologies and environments that facilitate the development of maintainable and verifiable software. A critical determinant in the design of tools to aid the software engineer is how the software engineering activity itself is conceptualized. Programming has traditionally been regarded as a problem solving activity. Thus, software engineering emphasis has often been on developing software aimed at solving particular classes of problems, even in cases where the applications do not fit our idea of a problem.

In contrast to this view of programming as problem solving, there is a novel way of viewing both the structure of programs and the structure of software engineering. Specifically, a computer program can be viewed as the specification of a process. The idea that a program specifies a process does not seem radical and may seem implicit in certain types of programming such as simulations. Nevertheless, it represents a shift in emphasis from the tradition of a program as describing a problem solving method. Processes may be problem solving activities, but they may be many other things, including real-time control and operating systems. For instance, many of the examples cited as "components of complex systems" by Winograd (1979) are most easily conceptualized as processes or as behavioral descriptions of systems, rather than as problem solving activities.

Once the model of programs as processes is accepted, the software engineering researcher's task becomes one of studying and providing facilities for the specification of processes in a disciplined and natural manner. Since KAT is specifically designed to provide an environment for the acquisition of process knowledge, it should be most helpful in this regard. KAT is being built using important principles of software engineering, but it is also clear that the techniques embodied in KAT for knowledge engineering can be fed back into the design of programming environments for software engineering.

If one thinks of programming as process specification rather than problem solving, the techniques developed in KAT for the acquisition of process knowledge are available for use in creating good programming environments. The basic entities by which processes are described in KAT include the lowest level subprocesses and the structures that are referenced in the description of the process at any level. In using KAT to develop software specifications for programming a process, natural language descriptions of subsystems known to be programmable would constitute these lowest level items. If KAT were being used to produce code directly, there would be statements in a programming language or modules already available.

There should be present in the software engineering environment a variety of statements of a functional, imperative, and declarative nature that can perform the desired operations. It should be possible to create new ones within the environment by compositions of processes and by definition of basic processes on data structures.

Additional basic entities in software engineering are the data structures used in programming. In current software engineering methodology, they are described by data abstraction techniques. In fact, this is exactly how their counterparts in knowledge engineering are described. For instance, in a navy knowledge-based system in which reference is made to ships, the system only knows what a ship is in terms of the operations that can take place on ships and the predicates that can be used to make queries about ships. A ship is a different data object in a system for naval personnel records than in a system for naval battle management.

2. Specific Areas of Tool Application

Suppose that the whole process of creating software for a particular application were developed using a tool such as KAT. At the very least, one could use the tool for recording and refining specifications for the system, and project assignments could be imposed over this structure. The description of the system could be developed by individuals or software engineering teams, down to the level of code. Specification, code, and documentation could be kept together, thus simplifying the project librarian's job.

In addition, one could use KAT's facilities for creating AND/OR and flowchart representations of a process to add to the documentation. Further, one might imagine that KAT's Prediction Mode queries such as "What could go wrong here?" and "How could the process be redesigned to avoid this problem?" would be helpful in fleshing out program content, making programs "correct by design". Finally, the ability to "step through" a program in KAT's Diagnosis Mode would provide an ideal facility for debugging.

The ways in which a tool such as KAT might be applied to software engineering efforts extend far beyond these specific examples. Below are listed seven generic problem areas of concern to software engineers that could be aided through techniques developed for knowledge engineers:

a. Code Design

This is the primary issue addressed previously. Generation of code could take place automatically through the specification of processes, just as knowledge bases can be automatically generated with KAT.

b. Project Design

In a large software project, the organization of the project reflects, to some degree, the organization of the system. How better to determine system segmentation than through gathering the expertise of persons with knowledge of the process for which the software is being designed? Once this information about the hierarchical project design is obtained, it can be made available to those managing the project who need a big picture view. Given an appropriate knowledge acquisition tool with a knowledge base that can be consulted centrally, managers and domain experts alike can add comments pertinent to the project design.

c. Interteam and Intrateam Communication

Any sort of computerized tool for project segmentation can facilitate communication within and among programming teams. This is analogous to the sharing of opinions among experts and AI programmers through the use of a knowledge engineering tool such as KAT.

d. Problem Diagnosis

As mentioned earlier, a facility such as the Diagnosis Mode serves a very important debugging function. Such a feature would provide programmers with an opportunity to observe the consequences of having specified the program flow in a particular way, and to embellish or redesign as necessary.

e. External Documentation

Imagine that one could couple knowledge acquisition as it continues through the development project with an expert system shell such as the KEE system used in KAT. It would then be possible to develop effective reference systems that anticipate and react to a user's needs, rather than producing canned documentation. This would be useful for many purposes, including tutoring new users.

f. Redesign and Reuse

By archiving the knowledge concerning the software design process, it should be possible to facilitate the maintenance, redesign, and reuse of the modules in the program. The importance of internal documentation to this process has often been remarked. The sophisticated history gathered by KAT should be doubly useful in this process. Representational aspects of KAT, including class hierarchies and instance variables, tend to promote reuse and redesign. As Fischer (1987) points out, inheritance allows simple redesign by allowing new objects similar to ones previously defined to be created with only a few incremental changes. Inheritance also reduces the need to specify redundant information, and simplifies updating and modification.

g. Testing and Verification

As an aid to software engineering, it has long been considered useful to provide interpreters that can simulate the execution of code in a production environment, thus helping to test that code. There is also the possibility of verification of the code, either heuristically, with relevant "expertise" built into the software engineering system, or through mathematical techniques such as the inductive assertion method (e.g., Floyd, 1967; Hoare, 1969).

In summary, KAT is designed for use in knowledge engineering, specifically in acquiring expert knowledge about processes. If one considers computer programming as the specification of processes, KAT has excellent potential as a software engineering tool as well. As the development of KAT progresses, BDM will continue to explore application areas other than materials processing where it might be usefully employed.

REFERENCES

1. Bobrow, D. (1985) Qualitative Reasoning about Physical Systems, MIT Press, Cambridge, Mass.

2. Caputo, A. J., and Lackey, W.J. (1984) Fabrication of fiber-reinforced ceramic composites by chemical vapor infiltration, Ceramic Engineering Sciences Proceedings, 5, 654-667.

3. Caputo, A. J., Lackey, W. J., and Stinson, D. P. (1985) Development of a new, faster process for the fabrication of fiber-reinforced ceramic composites by chemical vapor infiltration, Ceramic Engineering Sciences Proceedings, 6, 694-706.

4. De Kleer, J. (1984) How circuits work, Artificial Intelligence, 24. Reprinted in Bobrow, 1984.

5. Fischer, G. (1987) Cognitive view of reuse and redesign. IEEE Software, 4, 4, 60-72.

6. Floyd, R. W. (1967) Assigning meanings to programs. Proceedings of the American Mathematical Society Symposia in Applied Mathematics, 19, 19-31.

7. Graf, P., and Schacter, D. (1985) Implicit and explicit memory for new associations in normal and amnesic subjects. Journal of Experimental Psychology: Learning, Memory, & Cognition, 11, 501-518.

8. Hamilton, M., and Zeldin, S. (1976) Higher-order software--A methodology for defining software, IEEE Transactions on Software Engineering, SE-2, 1, 9-32.

9. Harel, D. (1980) AND/OR programs: A new approach to structured programming, ACM Transactions in Programming Languages and Systems, 2, 1, 1-17.

10. Hoare, C. A. R. (1969) An axiomatic approach to computer programming. Communications of the ACM, 12, 10, 576-580.

11. Kolers, P. A., and Roediger, H. L. (1984) Procedures of mind, Journal of Verbal Learning and Verbal Behavior, 23, 425-449.

12. Kuipers, B. (1984) Commonsense reasoning about causality: Deriving behavior from structure, Artificial Intelligence, 24. Reprinted in Bobrow, 1985, 169-204.

13. Reeker, L. H. (1986) Extended AND/OR graphs and parallel programming in a graphic environment, Proceedings of the International Computing Symposium 1986, Tainan, Taiwan, December, 138-146.

14. Roediger, H. L., and Blaxton, T. A. (1987) Retrieval modes produce dissociations in memory for surface information. In D. S. Gorfein and R. R. Hoffmann (Eds.) Memory and Learning: The Ebbinghaus Centennial Conference, Erlbaum, Hillsdale, N.J.