Perspectives on Knowledge in Engineering Design

by

William J. Rasdorf, P.E.

ABSTRACT

Several research organizations are working on the application of artificial intelligence (AI) techniques to engineering processes and operations (1, 3, 4, 5). Of all the contributions of AI, expert systems show some of the most significant promise for engineering disciplines. An expert system provides a framework for acquiring, representing, and using knowledge about a particular application's domain. This computer processing of knowledge is increasingly significant in a wide variety of engineering applications. Although the construction and processing of the knowledge bases of expert systems are currently widely analyzed, the role of knowledge in engineering design merits closer attention so that AI-oriented computer-aided engineering (CAE) systems can be developed and maintained systematically.

As in many current AI systems, knowledge processing is the most important issue in expert systems. "Knowledge" in engineering applications is loosely defined and subject to interpretation. Therefore, before widespread engineering design applications can be achieved, we must formally identify knowledge types and the correlation between them. To begin this process, this paper presents various perspectives of the knowledge currently used in engineering design.

INTRODUCTION

One of the most rapidly developing subdisciplines of computer science is artificial intelligence. AI seeks the exhibition by computers of behavior which would be called intelligent if it were being exhibited by humans (22). Of particular concern in AI is an understanding of the knowledge-processing capabilities needed to produce intelligent behavior. Such behavior includes organizing knowledge, exploiting constraints, controlling attention, searching solution spaces, logic, and theorem proving (22). Each of these behavioral methods is applied to the major areas of AI research: vision and image processing, speech, natural language processing, strategic technology, robotics, and expert systems, the latter of which is the topic of this paper.

Expert systems embody methods of conceptualization and reasoning to simulate human problem-solving capabilities in specific domains, e.g., engineering design. Expert systems rely on the acquisition, representation, processing, verification, and use of knowledge. Since knowledge is the central theme, expert systems are also called knowledge-based expert systems (KBES). The application of KBESs in engineering appears to be most promising (1, 3, 16, 19, 20).

In an AI-oriented CAE application, knowledge appears in many contexts. The primary ones are knowledge types, knowledge control, and knowledge representation. The interrelationships between these contexts and the criteria one uses to choose appropriate representation paradigms are discussed herein.

BACKGROUND

An expert system consists of three main components: an inference engine, a knowledge base, and a representation of the problem at hand. The inference engine is a computer program that uses knowledge and a representation of the problem to draw conclusions. The knowledge base is a collection of facts, procedures, and rules (or heuristics) gathered from an expert by a knowledge engineer. The expert is a person chosen for his or her level of performance in the domain being captured, for example, structural analysis. The knowledge engineer specializes in extracting information from the expert through interviews, role-playing, and observation. The knowledge engineer then selects tools and develops strategies to represent the decision processes of the expert in a computerized form.

Expert systems limit themselves to domain-specific knowledge rather than general problem-solving techniques. SACON, for example, dealt with structural analysis, using finite element techniques to simulate the behavior of a physical structure under various loading conditions. The more limited, or domain-specific, the problem, the greater its chances of being captured successfully in an expert system.

TYPES OF KNOWLEDGE

The most important and time-consuming task in the construction of KBESs is establishing a knowledge base through problem analysis and the acquisition and representation of the knowledge needed for problem solutions. In a variety of domains, the emergence of expert systems points out inconsistencies between knowledge sources and gaps in domain knowledge. These problems are prominent in engineering design, construction, and manufacturing processes where knowledge is not monolithic and information flow is fragmented. However, the construction of engineering knowledge bases can be greatly simplified by classifying knowledge according to its function. The types of knowledge commonly encountered in engineering design include:

- Facts,
- Procedures (algorithms, operations),
- Judgments (heuristics, beliefs), and
- Control.

<u>Facts</u>

Factual knowledge is the most primitive, declarative knowledge type. It consists of formalized knowledge typically found in handbooks. Material specifications and engineering properties contain factual knowledge. Additional candidates for factual representation in mechanical design applications include:

- ASTM standards,
- AISC tables,
- Piping handbooks,
- Material properties,
- Equipment specifications, and
- Engineering data.

Factual knowledge is highly amenable to database management system processing (16).

In expert systems, factual knowledge has most successfully been used in R1, a KBES which configures computer orders to the specifications of clients (13). The factual knowledge base of R1 contains the engineering specifications and sizes of computer hardware. This factual knowledge base is used in conjunction with the decision making knowledge of R1 to configure Digital Equipment Corporation VAX computer systems.

Procedures

Procedural knowledge is by nature algorithmic and operative. Algorithms constitute either numeric or non-numeric step-by-step procedures for solving a problem or accomplishing some end. In general, algorithms cause a transformation of facts from one state to another. Engineering algorithms are more generally numeric. They are founded on the methods of algebra and calculus, i.e., they are analytic. Analysis algorithms, for example, transform geometric and external loading data to internal component forces and moments using the laws of equilibrium.

Operative knowledge processes factual knowledge; it creates, deletes, modifies, and transports facts. It also provides a flexible interface between an engineer and the computer as exemplified by problem-oriented languages (POLS) and natural-language interfaces.

Some examples of application programs that embody procedural knowledge include:

- Finite element analysis,
- Database management system,
- Optimization, and
- Graphics programs.

Procedural knowledge, then, conveys how data transformations and operations are to be achieved.

Judgments

Judgmental knowledge cannot easily be coded into traditional computer programs. It consists of beliefs, which embody an individual's convictions and opinions, and heuristics, which include observations, judgment, experience, rules of good practice, plausible reasoning, and knowledge of widely accepted procedures and methods. Judgmental knowledge relies on logic, the formal principles of reasoning, to provide a framework for its use.

Undeniably, judgmental knowledge is not a type that engineers traditionally encode in computer programs. Yet judgmental knowledge is essential to engineering design processes. In fact, judgmental knowledge plays a major role in all phases of the design process: conceptualization, requirement definition, planning, preliminary and detailed analysis, proportioning, sizing, procurement, manufacturing, operation, testing, and revision. The engineer uses judgment to evaluate alternatives and to guide the search for design solutions. Judgmental knowledge is quite often cast as regulatory requirements and is embedded in codes, specifications, standards, and other regulatory documents.

Control

Control knowledge is metaknowledge, knowledge about knowledge. Control knowledge directs the processing of the above types of knowledge in expert systems. Control knowledge can be procedural, judgmental, or both. Important features of control knowledge include:

- Pattern-directed actions,
- Flexible and sometimes self-modifying behavior,
- Anticipation of unexpected developments,
- The use of an open-ended solution space, and
- Dealing with uncertainties.

Control knowledge controls and coordinates the evolution of the other knowledge forms by choosing appropriate knowledge sources for a particular situation. Depending on the level of intellect and the required expertise, control knowledge can be categorized into three classes. The next section discusses these classes and presents pertinent examples.

CLASSES OF KNOWLEDGE

The knowledge used in engineering is not uniform in its nature and applicability. It can be at a very high level, where general decisions are made; it can be very specific, concerning a single component or a simple activity; or it can assume varying degrees of complexity between these two extremes. Chandrasekaran (1, 3) categorizes the design tasks of an industrial process in similar terms. He identified three classes of design tasks that depend on whether the task is creative, innovative, or routine. These categories are applied herein to the various levels of engineering design knowledge.

Creative knowledge involves a high level of attention in a broad sense. It presents the engineer with differing and often contradictory options, particularly in initial design stages where conflicts between goals must be resolved and resources must be conserved, and during critical phases of product development where unanticipated problems arise and contingency plans must be put into effect. Engineers use creative knowledge to select a site for a manufacturing plant, initiate a new product line, evaluate initial design options, and change the marketing strategy of a product.

Innovative knowledge is not as general as creative knowledge, but still involves considerable decision-making effort and is used more or less routinely throughout the design/manufacture process. Engineers use innovative knowledge to select computer hardware or software for a particular application, manage and schedule projects, fine-tune a process to match manufacturing specifications, and design a facility given the loads, materials, and other constraints.

Routine knowledge enables one to perform the day-to-day tasks of engineering. This is not to say that routine knowledge does not warrant any expertise or skill, but that it is less general than creative or innovative knowledge. At this level, decision-making ingredients are well understood and engineering alternatives are chosen from a large, but well-defined, knowledge base. Engineers use routine knowledge to diagnose a specific malfunction, optimize a process, control and monitor activities, and check designs.

These classifications of engineering design knowledge are not rigidly definable. The levels presented tend to overlap one another to some extent and could themselves be further subdivided. However, the above classification sufficiently illustrates the variety of knowledge needed to perform different engineering design tasks.

EXPERT SYSTEM/ENGINEERING TASKS

Having discussed the types and classes of engineering design knowledge, this section will now define feasible tasks for expert systems before covering the selection of KBES tools for various engineering tasks and applications. The tasks fall into two broad categories, interpretive tasks and generative tasks.

Interpretive tasks require reasoning about a task in light of the knowledge available in the task's domain. Interpretive tasks are data driven. They obtain known data, reason about it to make decisions and inferences, and then derive problem solutions. Examples include interpretation itself, diagnosis, monitoring, and prediction.

Generative tasks are those for which there are many possible outcomes. A solution to a generative problem is obtained by generating a potential solution and testing it against a set of applicable constraints. Generative tasks are goal driven. They establish a goal, seek solution methods to achieve the goal, and then apply constraints to limit the solution space. Examples include planning and design.

Brief definitions of these tasks, including examples, follow (8).

Interpretation - Analyze known data to determine its meaning. Example: the interpretation of laboratory test data.

Diagnosis - Investigate the cause of a fault in a system. Example: the diagnosis of a structural failure.

Monitoring - Continuously interpret signals and notify users when intervention is required. Example: real-time monitoring of stress and strain parameters in a structural system test.

Prediction - Forecast the future from a model of the past and the present. Example: predicting the effects of time-dependent changes in material properties.

Planning - Create programs of action to achieve a goal. Example: planning the manufacturing sequence for an assembled part.

Design - Develop specifications to create objects that satisfy a set of constraints. Example: the design of a component, part, or facility.

Other suitable tasks for expert systems include consultation, instruction, evaluation, scheduling, debugging, testing, and repair. Clearly, one must recognize the distinguishing characteristics of these tasks to effectively apply KBES tools to them.

SELECTION OF KBES TOOLS FOR ENGINEERING APPLICATIONS

Applying KBES techniques to engineering problems requires classification and organization of the knowledge levels involved, followed by a consideration of appropriate tools for developing and implementing a KBES. Many factors contribute to deciding what criteria enable a KBES tool to meet engineering design requirements. Magliero presented the following as critical features of expert-system tools (9):

- Knowledge representation techniques,
- Control strategy,
- KBES interface requirements,
- Compatibility with traditional application programs, and
- Economic considerations.

Those considering the acquisition of KBES tools must evaluate these features carefully. Each feature will be briefly discussed below. For a broader discussion and a comparison of popular expert-system tools based on these criteria, the reader is referred to the paper by Magliero.

Knowledge Representation Techniques

Knowledge-based paradigms are the techniques used to represent and process the knowledge of expert systems. The four most widely used paradigms are semantic networks (5, 18), production rules (7, 13), frames (11, 12), and first-order predicate logic (4, 21). Although currently most expert system tools support only one knowledge representation and processing paradigm, it is possible to develop tools which support multiple knowledge paradigms. A number of expert systems have been developed, for example, using a combination of production rules and frames (10, 17, 23).

Control Strategy

The control strategy processes domain knowledge. Although control knowledge is separate from domain knowledge, the implementation of a control strategy depends heavily on how knowledge is represented. Knowledge paradigms, therefore, often directly determine control strategies. A single type of knowledge representation paradigm can employ different kinds of control strategies although the reverse is not true. For example, the production rule paradigm of EMYCIN (2) follows a goal driven strategy, whereas the production rule paradigm of OPS5 (7) follows a data-driven strategy.

Typical planning-analysis-design cycles of engineering activities mix goal- and data-driven strategies. Therefore, an expert-system tool having control knowledge that employs different knowledge-processing techniques would be ideal for engineering applications.

KBES Interface Requirements

Ease of use contributes to the ultimate utility of any major software tool. In the case of a KBES, such use involves three activities: knowledge acquisition, knowledge maintenance, and explanation.

Like many typical engineering and data-processing activities, knowledge acquisition and maintenance are cyclic, requiring definition and redefinition as experience is gained. During knowledge acquisition a knowledge engineer extracts knowledge from an expert and encodes it in a KBES. The KBES interface should make this high level activity as easy as possible by providing capabilities that free the knowledge engineer and the expert from software-dependent details.

Initial attempts to encode knowledge obtained from experts may result in flaws or contradictions in the knowledge base. In addition, modifications during the lifetime of the expert system can introduce incompatibilities and contradictions. Remedial actions on the knowledge base are therefore necessary. These actions fall within the realm of knowledge maintenance, which involves the addition, deletion, and modification of knowledge. An appropriate debugging facility is therefore essential.

Another important requirement that must be met by the KBES interface is an explanatory capability. The expert system must be able to explain its reasoning process. An explanatory capability aids in detecting flaws in the system during development. It also aids novice users in understanding the expert reasoning process of the domain being modeled. Lastly, it pursuades human experts of its validity by revealing the reasoning leading to its conclusions.

Compatibility With Traditional Application Programs

One of the major impediments to the use of KBES tools by engineers is the languages in which they are written. More often than not these languages have been LISP and PROLOG, although other languages, including FORTRAN, have been used (15). Because of a lack of standardization, these languages have not provided a uniform development base, hindering their use by engineers.

The importance of compatibility in a computer-aided engineering environment cannot be overemphasized. However, consideration of the compatibility between the broad spectrum of tools available to the engineer, including application software, database management systems, and knowledge-based expert systems, is often neglected. It is imperative, therefore, that an efficient interface between expert system tools and traditional engineering software be investigated to provide an environment supportive of CAE departments in all of these areas. Those tools which most easily fit into an engineering organization's computing environment must be identified as high priority candidates for acquisition.

Economic Considerations

The cost effectiveness of a KBES application requires careful study. In most cases, the cost of the hardware and software needed for expert system development will be low compared to the cost of knowledge acquisition, encoding, and maintenance. Since the integration of KBES and traditional engineering computing tools is significant, the availability and cost effectiveness of such an interface should be examined closely. The ease with which such tools could fit within an organization's computer-aided engineering environment is a strong consideration. As when acquiring any major software, engineering organizations need to evaluate the cost of KBES relative to other traditional engineering applications, then base their acquisition decisions on the economic impact this new technology would have on their current and future operations.

SUMMARY

Expert systems is a rapidly developing subdiscipline of artificial intelligence that simulates human problem-solving capabilities. Constructing an expert system often points out inconsistencies in domain knowledge while formalizing it. The types of domain knowledge; facts, procedures, judgments, and control; differ from the classes of that knowledge; creative, innovative, and routine. The feasible tasks for expert systems can be determined based on these types and classes of knowledge. Interpretive tasks require reasoning about a task in light of the knowledge available, where generative tasks create potential solutions to be tested against constraints. Only after classifying the domain by type and level can the engineer select an appropriate knowledge-engineering tool for the domain being considered. The critical features to be weighed after classification are knowledge representation techniques, control strategies, interface requirements, compatibility with traditional systems, and economic considerations.

ACKNOWLEDGEMENTS

Portions of this work were sponsored by the Integrated Manufacturing Systems Engineering Institute of North Carolina State University under a grant entitled "Information Acquisition, Representation, and Management in the Design and Control of CIM Systems" and by the National Aeronautics and Space Administration under a grant entitled "Data Management Technology - Functional Requirements for a Generative Database." This paper is based in part on a preliminary report co-authored by Mr. Josyula V. Sitaram. The author wishes to acknowledge the significant contributions of Mr. Sitaram.

REFERENCES

(1) Brown, D. C., and Chandrasekaran, B., "An Approach to Expert Systems for Mechanical Design," Proceedings, Trends and Applications 1983, National Bureau of Standards, Gaithersburg, MD, May 1983.

(2) Buchanan, B. G., and Shortliffe, E. H., Rule-Based Expert Systems, Addison-Wesley Publishing Company, Reading, MA, 1984.

(3) Chandrasekaran, B., "Expert Systems: Matching Techniques to Tasks," Annual Report for Research on Distributed Knowledge Base Systems for Diagnosis and Information Retrieval, Air Force Office of Scientific Research, Bolling Air Force Base, Washington, DC, November 1983.

(4) Dahl, V., "Logic Programming as a Representation of Knowledge," COMPUTER, IEEE Computer Society, Volume 16, Number 10, October 1983.

(5) Duda, R. O., "The Prospector System for Mineral Exploration," Final Report, SRI Project 8172, Artificial Intelligence Center, SRI International, Menlo Park, CA, April 1980.

(6) Erman, L. D., et al., "The Hearsay-II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty," Computing Surveys, Volume 12, Number 2, Pages 213-253, June 1980.

(7) Forgy, C. L., "The OPS5 User's Manual," Technical Report CMU-CS-81-135, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA, July 1981.

(8) Hayes-Roth, F., Waterman, D. A., Lenat, D. B., Building Expert Systems, Addison-Wesley Publishing Company, Reading, MA, 1983.

(9) Magliero, A., "Review of Expert System Tools for a Prototype Project," Software and Architecture and Engineering, Inc., Arlington, VA, 1984.

(10) Maher, M. L., and Fenves, S. J., "HI-RISE: A Knowledge-Based Expert System for the Preliminary Structural Design of High-Rise Buildings," Technical Report R-85-146, Department of Civil Engineering, Carnegie-Mellon University, Pittsburgh, PA, January 1985.

(11) Minsky, M., "A Framework for Representing Knowledge," The Psychology of Computer Vision, edited by Winston, P., McGraw-Hill, New York, NY, 1975.

(12) Mittal, S., Chandrasekaran, B., Stricklen J., "Patrec: A Knowledge Directed Database for a Diagnostic Expert System," COMPUTER, IEEE Computer Society, Volume 17, Number 9, September 1984.

(13) McDermott, J., "R1: A Rule-Based Configurer of Computer Systems," Artificial Intelligence, September 1982.

(14) Nau, D. S., "Expert Computer Systems," COMPUTER, IEEE Computer Society, Volume 16, Pages 63-85, February 1983.

(15) Holtz, N. M., and Rasdorf, W. J., "LISP - A CAD System Programming Language," Journal of Technical Topics in Civil Engineering, American Society of Civil Engineers, Volume 109, Number 1, Pages 58-72, April 1983.

(16) Rasdorf, W. J., and Salley, G. C, "Generative Engineering Databases - Toward Expert Systems," Computers and Structures, Spring, 1985.

(17) Rauch-Hindin, W., "Expert System to Plan PC Board Assembly in the Factory," SYSTEMS and SOFTWARE, Volume 3, November 8, August 1984.

(18) Reiter, J. E., "A Review of the Prospector Project," Data Base Engineering, Quarterly Bulletin of the IEEE Computer Society Technical Committee, Volume 6, Number 4, December 1983.

(19) Rychener, M. D., "Expert Systems for Engineering Design: Problem Components, Techniques & Prototypes," Technical Report, Design Research Center & Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, March 1984.

(20) Sriram, D., et. al., "Applications of Expert Systems in Structural Engineering," Proceedings of the Conference on Artificial Intelligence, Oakland University, Rochester, MI, April 1983.

(21) Swinson, P. S. G., Pereira, F. C. N., and Biji, A., "A Fact-Dependency System for the Logic Programmer," Computer-Aided Design, Butterworth and Co., London, Volume 15, Number 4, July 1983.

(22) Winston, P. H., Artificial Intelligence, Addison-Wesley Publishing Company, Reading, MA, 1984.

(23) Wright, J. M., and Fox, M. S., "SRL/1.5 User Manual," Technical Report, Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, June 1983.