KNOWLEDGE-BASED TECHNOLOGY FOR TRAINING SYSTEMS

Dr. Matt Narotam
Mr. Gary Hassett

BURTEK
Tulsa, Oklahoma

ABSTRACT

Expert systems can potentially be used to reduce the cost of building and maintaining training systems. As expert system technology has grown, several tools have become available for building expert systems. The tools are hosted on low-cost work stations or personal computers and provide varying degrees of sophistication for user interface and knowledge representation.

Before, we can rush out and mass-produce useful expert systems, factors that impact the development of expert systems must be investigated. Some of these factors include determining the steps involved in the knowledge engineering process, the qualification criteria for knowledge engineers, performance and limitations of tools available for developing the knowledge-based systems, and development of strategies for integrating the expert systems in the engineering or product environment. These issues are nontrivial and numerous.

This paper begins to attack the problem by presenting experience gained and lessons-learned from a project that involved building an expert system. The objective of the project was to focus on issues relating to the knowledge engineering process, especially that of knowledge retrieval. Therefore, the expert system was developed to solve a relatively simple problem - determine the cause of the malfunction of a modem system.

INTRODUCTION

This paper presents the results of an Artificial Intelligence/Expert System (AI/ES) project. Several aspects of this project are discussed. First, the project was intended to investigate issues involved in the use of AI/ES technology and investigate its applicability to reduce overall training system life-cycle cost. The factors that impact this cost are reasonably well known. The unknown quantity is the emerging technology of artificial intelligence and expert systems. To inject the required technical expertise for this project, Burtek decided to draw on additional resources available externally.

The State of Oklahoma, in seeking to diversify its economy from dependency on mainly oil and agriculture, has elected to invest in applied research activities relating to this technology. The initial investments will be in the form of grants made available to support Oklahoma's high technology industry.

This state-level involvement has triggered technical activities that contributed to this project. The first such activity is in the form of resources and support provided by the University of Tulsa's Mathematics and Computer Science Department. In addition, the University of Oklahoma, using State and Industry funds, sponsored a six-month program called Base of Expert System Technology (BEST) of Oklahoma, which included training, development of an application, and exchange of information and experience between participants. Burtek is a participant in the BEST program, with a project entitled: "Technology Insertion for Maintenance and Support of Simulators."

GOALS OF THE PROJECT

The goals of the project were first to obtain a better understanding of expert system technology and how it can be used for reducing the cost of building and maintaining training systems. As with any emerging technology, there has been tremendous exaggeration of the benefits of AI/ES, which has resulted in a great deal of management skepticism and user disappointment.

In addition, information of substance is difficult to obtain because of sensitivity concerning specific applications, exacerbated by the lack of such information. Therefore, another goal of the project was to build a "demonstrable product."

EXPERT SYSTEM TECHNOLOGY

Expert system technology has become more "affordable." Expert system shells available for PCs and workstations provide varying degrees of sophistication in the areas of user interface, knowledge representation, "reasoning" implementation, and interfaces to languages such as "C" and LISP. They are usable and flexible, and we are encouraged to consider them as "nonprogramming" tools, i.e., more as data base management tools. As a technology however, they provide the capability to build knowledge-based tools.

Once it has been determined just what is the knowledge-base, then there are several issues concerning expert system technology that must be addressed.

- What is the usefulness of knowledge-based technology?

- What are the cost issues?

  - Engineering
  - Training

- How can the technology be inserted into an engineering or support activity?

- Why and where should the technology be used?

- Are there qualified personnel available?

- What are the training issues?

- Are experts willing to help?

This paper is not intended to answer all of these questions but to provide information that would spur additional studies to be performed.

PROBLEM SELECTION

In selecting a problem for implementation, consideration was given to several requirements:

- It was understood that the knowledge engineering process in Expert System development was most important.

- The project must demonstrate the usefulness of expert system shells.

- The product must demonstrate the application of the technology.

- The product must be usable and acceptable.

To meet these requirements, the following criteria were established:

- The problem must be of reduced complexity. Problem: how to define or measure complexity? Rules? Lines of Code?

- It was essential to focus on knowledge engineering issues. This meant that a development plan had to be established that allowed sufficient consideration of knowledge engineering techniques.

- It was essential to provide an environment for success.

- We should not be constrained by implementation tool shortcomings.

- We needed "friendly" domain experts.

Several problems were considered. Some of these were:

- Computer selection process (for training systems)

- Software/hardware troubleshooting problems

The process of selecting computers for training system procurements typically involves a number of experts. Since a goal of this project was to develop a successful "product," this project was considered too complex to implement in a short period of time. A decision was made to simplify the problem and target a single subject matter expert. Another decision was to select a problem that could be easily demonstrated. The result was to select a recent, accessible, and well-known hardware diagnostic/troubleshooting problem.

Employing the above criteria, the following *problem was defined:* **Capture the *expertise of an* individual who troubleshoots modem problems. The expert system would be used for determining the cause of modem-use problems.**

At this point, it must be noted in retrospect, no consideration was given to the "user" of the system. We understood the problem to be more a knowledge capturing and representation exercise. As we shall see later, the "user" is very much a part of the knowledge engineering process.

An additional consideration was the constitution of the development team. Typically, a team would consist of a subject matter expert and a knowledge engineer.

Unfortunately, some people who call themselves "knowledge engineers" are no more than people with *some working knowledge of expert system tools. In* addition, certain vendors of expert system shells make claims that the tool is sufficiently user-friendly to allow the domain expert to be the knowledge engineer.

As will be shown later, an understanding of reasoning paradigms and knowledge representation is important. But equally important is the understanding of the process necessary to engineer the knowledge-based product.

A very real problem is the lack of good knowledge engineering training, and the lack of relevant information in papers and articles.

THE DEVELOPMENT ENVIRONMENT

The "Intelligence/Compiler" is an Expert System development environment and knowledge compiler which allows one to:

- Develop knowledge-bases using an intelligent editor

- Combine several knowledge representation methods

- Compile knowledge for execution

The compiler is installed on an IBM PC and allows:

- The combination of several programming methods and paradigms within a single environment.

- The interleaving of Artificial Intelligence and conventional programs.

Intelligence/Compiler provides a uniform framework for representing knowledge with multiple paradigms, such as rules, frames, or logic, and implements a compilation mechanism which allows all of these paradigms to be freely mixed, compiled, and executed. **Intelligence/Compiler** also provides an intelligent editing and development environment which manages knowledge structures and their interrelationships.

Intelligence/Compiler provides facilities for:

- Rule-Based Programming: in which knowledge is represented as a set of if-then rules with logical premises and conclusions. **Intelligence/Compiler** supports forward chaining, backward chaining, and inexact reasoning rules which may be freely combined.

- Frame-Based Programming: which relies on structured knowledge for capturing regularly occurring circumstances. Each frame includes a number of slots with inheritance, predicate attachments, and active values.

- Logic-Based Programming: which deals with logical predicates and assertions. **Intelligence/Compiler** includes a logical knowledge representation system which combines forward and backward chaining inference with backtracking and cuts.

- Procedural Programming: in which **Intelligence/Compiler** provides interfaces to "C" programs. Any predicate in "Intelligence/Compiler" may involve a call to a conventional "C" function and may pass or receive values from "C."

DEVELOPMENT PROCESS

Several references (1, 2, 3) were reviewed to obtain information that could be used to establish a development plan. For the purpose of this project, four phases of activities were planned.

- Problem Definition Phase
- Knowledge Acquisition Phase
- Knowledge Extraction and Prototyping
- Knowledge Base Refinements and Implementation

Problem Definition Phase

During this phase, time was spent understanding the problem being solved and establishing a project plan. The steps of this phase include identifying the specific task to be implemented, specifying functions to be performed, and determining the knowledge-base requirements. The problems encountered during this phase stemmed from insufficient experience with both the knowledge engineering process and the application of expert system technology. Consideration of where best to apply the technology and how to demonstrate the potential benefits served to confuse the issues. Considerable progress was made when a decision was made to select any "small" problem situation and develop an expert system for that problem.

Knowledge Acquisition

This phase of the project involved interviewing the domain expert, recording the interview, and generation of a transcript.

One of the problems encountered was determining the length of time for the interview. Huge volumes of transcript material can be produced for lengthy periods of discussions. This would pose an information management problem and in addition, having produced the material, it has to be read and analyzed. It was established that in an interview session, as time progressed, less but useful "information" is generated. In terms of returns, *i.e., rules extracted per hour, more effort* is required for production of rules as more information is analyzed. This phase of the project really is one of getting to know the problem better. There is some cutoff point where "enough" is known about the problem. Alternative methods of knowledge acquisition involve observing the expert at work (4).

Knowledge Extraction and Prototyping

During this phase of the project, the interview transcript was analyzed to obtain facts and rules. The domain expert was consulted to verify facts and obtain new information. The facts and rules were converted into knowledge representations (5, 6). An expert system solution was structured by means of prototyping. This and subsequent phases of the project were influenced by results of prototypes that were developed. This influence was mainly as a result of considering how the "user" interfaces with the expert system product.

Knowledge-Base Refinement and Implementation

As a result of initial prototyping, further analysis and discussions were held with the subject matter expert that led to a refinement of the knowledge-base extracted from the interview transcript. A better understanding of the problem was obtained which involved targeting the product *for the user. The user was found to be a shifting* target. Additional "prototypes" were developed before a final "demonstrable" product was produced.

PROTOTYPES

There were three major expert system prototypes for the problem. As each was produced and tested, information was generated that led to reconsideration of knowledge engineering issues.

The knowledge acquisition phase results in identification of the following:

- Facts about the problem
- Operational knowledge
- Propositional knowledge
- Inexact or fuzzy knowledge

The difference between operational knowledge and propositional knowledge can be seen by considering the following:

- To determine if the modem power cable is connected to the power supply, one must check that the power cable is connected to the power supply. This is operational knowledge.

- To know that it can be determined that the power cable is connected by simply examining the power cable and observing that it is connected, is propositional knowledge.

The intent of this project has been to understand the knowledge-base technology. Therefore, one must understand knowledge and how (and whether) it can be used successfully.

From this project, much is revealed by *observing the performance of the prototypes. The* following is a discussion of general impressions of the performance of prototypes developed at various phases of the project.

The First Prototype

For the first prototype, a number of rules were implemented. In expert systems of this type, there is a need to interface with the user. The initial concept of the user interface was one that asked the user about facts concerning the problem. The degree of user interaction was not scoped.

The product appeared to gain all its "knowledge" from the user. Much of the burden for determining facts was placed on the user. The knowledge-base content of the product was almost nil. There were rules programmed, but the system "knew" very little. This led to the (harsh) question - where is the knowledge?

Additionally, the solution was very procedural. It was observed that this was the case because the construction of the solution was influenced by our procedural mind-set (developed for typical computational problem solving).

The Second Prototype

More "facts" or knowledge were installed into the solution. However, there was still a heavy reliance on the user to interpret information. Again, the system knew little. This resulted in a new perspective *on "knowing" and the need for some* sort of measure of knowledge content. It was felt that some structure was required to interface the user with the expertise. The solution was still very procedural.

The Third Prototype

It was determined that the use of objects and frame representations allowed the development of a better structured solution. The development approach involved identification of knowledge points or levels that allowed a network to be developed. This approach reduced the burden on the user and allowed greater insertion of knowledge into the system. This approach also "released" us from our procedural mind-set.

LESSONS LEARNED

The following is a discussion of issues and lessons learned during the project.

The Tool

The tool was reasonably good for the project, but the documentation was very poor. Once we understood how the framing and inheritance structure worked, they were simple to use. There is a great deal to be said for simplicity in expert system development. The editing facility was poor. Very often there was a requirement to reproduce structures which were not easily accommodated.

Knowledge Engineering

More published work is required in the area of knowledge acquisition and extraction, specifically in the identification of knowledge and representation of knowledge.

The user interface design is part of the knowledge engineering problem. The knowledge of the domain expert does not represent a solution. A good domain expert is typically knowledgeable about other matters. The end user of the product, typically, has a different knowledge set. For knowledge engineering applications, there is a need to:

- Scope the domain experts knowledge, i.e., obtain the pertinent knowledge set of the source.

- Establish the target (user) knowledge set.

- Engineer an interface or structure between the source and target knowledge sets.

An understanding and development of knowledge engineering and AI techniques is required for representation of propositional (facts or knowledge), the use of propositions to obtain new information (rules/reasoning), and the use of object-oriented representations (facts/objects).

A typical software engineering structure/process is not applicable for initial development of the expert system until basic procedures and techniques are well-understood.

Problem Application

Development of an expert system requires a good understanding about the application. The technology is best applied in well-known, specific problem situations. If the problem area is not well understood, then more time will be spent learning about the application area. This should be undertaken as a separate project.

Training of Developers

Training packages tend to focus on the expert system building tools. Our experience has led us to conclude that training must also incorporate basic AI and knowledge engineering techniques.

CONCLUSION

This project was designed to provide information relating to the application of knowledge-based technology for training systems. Several lessons have been learned as a result of this application. The focus has been on obtaining a better understanding of the activities that must be performed during the knowledge engineering process. The technology is advancing, especially in the area of tools and techniques. More tools with enhancements to user interfaces are being produced. Representation techniques such as neural networks are state-of-the-art. The results of this project are applicable to projects that will evaluate these new and upcoming technologies.

ABOUT THE AUTHORS

DR. MATT NAROTAM is a Staff Engineer at Burtek. His responsibilities include investigating the application of advanced technologies for building and maintaining training systems. Dr. Narotam has several years experience in system and software engineering. He was responsible for technical management of the USAF Ada Simulator Research project. In this role, Dr. Narotam played a principal part in the development of methods and techniques for system/software engineering. Dr. Narotam obtained his PhD from the University of Salford, England. His thesis was on "Investigation of Techniques for Continuous System Simulation."

MR. GARY HASSETT is Director of Engineering at Burtek. Mr. Hassett's organization includes functions for systems engineering, product design, and integrated logistics support. Mr. Hassett is responsible for enhancement and insertion of technologies that are employed in these engineering functions. Mr. Hassett has several years of experience in the development and maintenance of flight simulators as systems and Software Engineer, Project Engineer, and Program Manager. Mr. Hassett has a BSc in Aeronautical Engineering from Pennsylvania State University.

REFERENCES

1. James R. Geissman and Roger D. Schultz, "*Verification and Validation of Expert* Systems," AI Expert, p26-33, February 1988.

2. Paul Harmon and David King "Expert Systems - Artificial Intelligence In Business," John Wiley & Sons, 1985.

3. Wendy B. Rauch-Hindin, "Artificial Intelligence In Business, Science, and Industry, Vols I and II," Prentice-Hall, 1986.

4. Robert R. Koffman, "The Problem of Extracting the Knowledge of Experts from the Perspective of Experimental Psychology," AI Magazine, Publication of the American Association for AI, p53-67, Summer 1987.

5. Eugene Charniak and Drew McDermott, "Introduction to Artificial Intelligence," Addison-Wesley Publication Company, 1986.

6. Patrick Winston Henry, "Artificial Intelligence," Addison-Wesley, 1984.