1986

Practical issues for expert systems /

Carol A. Koperna
Lehigh University

PRACTICAL ISSUES FOR EXPERT SYSTEMS

by

Carol A. Koperna

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Computer Science

Lehigh University

1986

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

November 18, 1986
(date)

Donald J. Hillman
Professor in Charge

Donald J. Hillman
Division Head

Lawrence D. Varnin, Jr.
Chairman of Department

I wish to thank my employer, American Telephone & Telegraph Technologies, Inc., whose tuition refund program enabled me to pursue this degree, and also my husband Daniel, for his great faith and patience.

PRACTICAL ISSUES FOR EXPERT SYSTEMS

TABLE OF CONTENTS

I. ABSTRACT 1

II. Expert System Knowledge Representation 2

III. The Acquisition of Knowledge 13

IV. Technical Issues - Current and Future 22

V. MIS Staff - From Data Management To Knowledge Management 27

VI. Expert System Evaluation 32

VII. Expert Systems - Worth The Cost? 38

VIII. Integration With Traditional Software - A Must . 47

IX. Closing Comments 52

X. BIBLIOGRAPHY 55

XI. Personal Data 63

I. ABSTRACT

The information processing industry, its hardware, software applications, and services will be the world's most significant industry by the turn of the century. However, almost all of information processing activity is neither calculating or filing, which is what most computer systems do now, but reasoning. Expert systems present a real alternative to this problem. They offer a new way to address the age-old problem of transferring knowledge from the experienced to the inexperienced. Because of expert systems, knowledge can be viewed in a different way, no longer residing just in books or in the minds of a few experts, but accessible online, per users' need.

But there are major obstacles to the widespread use and acceptance of expert systems. These obstacles range from philosophical to technical and economic. There will always be people who refuse to believe that machines can think, who doubt that hardware and software will ever have sufficient capability, and that paybacks from the often significant investment required for expert systems will ever be realized.

This paper will explore some of the aspects of potential roadblocks and concerns of expert system development.

II. EXPERT SYSTEM KNOWLEDGE REPRESENTATION

The field of Artificial Intelligence is confusing because developments are proceeding at several different levels. At the upper end of the scale are the very ambitious projects designed to perform as real time tasks which are too complicated for humans to perform. At the lower end are expert systems able to advise on specific narrow domains at a technician or clerical level. In between these extremes is a spectrum ranging from major to medium-sized systems, requiring years of development and incorporating knowledge engineering on a substantial scale.

Expert systems are an aspect of Artificial Intelligence whereby problems will be solved not so much by data crunching as by heuristic processes akin to those of the human brain. Expert systems may be defined as intelligent decision support systems; computer programs which exhibit similar performance to human experts in performing tasks which normally require human intelligence. A system that could produce results which if produced by a human would be considered intelligent, can be thought of as expert. This simulation of people in the roles of experts in some topic is the closest we can get to cloning at this time. Expert systems have the potential to make widely available the working knowledge of an expert to those without direct access to the expert.

Although the first commercially successful expert systems did not appear until 1980, expert systems emerged in the fields of medicine and chemistry as early as the 1960's. Two of them, developed at Stanford were DENDRAL, for deducing the chemical structures of molecules, and MYCIN, for diagnosing infectious diseases.

The basic structure of an expert system is comprised of a knowledge base, inference engine, and user interface.

THE KNOWLEDGE BASE

The knowledge base contains all the necessary information to solve the problem at hand. It is obtained from experts who describe their tasks as a collection of heuristics. Experts tend to express most of their problem-solving techniques in terms of situation-action or IF-THEN rules. Therefore the most widely used type of expert system is the rule-based, where knowledge is represented in hundreds or thousands of IF-THEN rules and facts. The rules express a condition where the interpretation of the rule is that if the antecedent (IF part) can be satisfied the consequent (THEN part) can too. Facts are used to express assertions about properties and relations, and are generally static. Meta-rules are rules about the form and use of rules. While ordinary rules contain knowledge about the subject domain, meta-rules can contain information about the rules and search strategies in selecting paths of reasoning.

A variation of rule-based systems are example-based, popular with expert system shells. When the user enters actual examples, the shell infers the rules. IF-THEN rules are not the only way to represent knowledge. Frame-based structures use a prototype to describe a class of objects. The frames, similar to a record in a data file, have slots which are typical characteristics of the class of objects. A particular example of a class can be defined as an instance of the class.

Regardless of how a knowledge base is implemented, it is a necessity to represent the uncertainty of impreciseness and incompleteness of data. Uncertainty can be incorporated into a rule (frame, etc.) by providing a numerical value for the degree of certainty. In the development of MYCIN, it was necessary to encode rules that could not be encoded in terms of complete truth.[1] A numerical truth value was assigned to a rule and a way developed to combine truth values in chains of inference.

THE INFERENCE ENGINE

The inference engine is the logic structure of an expert system; the way it applies the information in the

---

[1] Bruce G. Buchanan and Edward H. Shortliffe (eds.), Rule-Based Expert Systems: The Mycin Experiments Of The Stanford Heuristic Programming Project, Addison-Wesley, Reading, Massachusetts, 1984, p. 91.

knowledge base to the problem until a final assessment is made. A forward-chaining or data-driven engine looks at the left-hand side of a rule (IF part) and determines whether it is true. If if is, the right-hand side of the rule (THEN part) is declared true. The rule is now considered to be fired, and the engine will search for rules whose IF side matches the THEN side of the rule just fired. This continues until no more rules can be fired. If more than one rule is true, the inference engine selects the first rule or perhaps selects on the basis of a rule priority.

A backward-chaining or goal-driven engine starts with a goal and works backward through the rules in the knowledge base considering only those rules whose THEN part contains the goal. When it finds one, it takes the IF part of the rule and looks for another rule that has the same terms in its THEN part until it has fired all rules necessary to prove the goal. If the goal cannot be satisfied, another goal can be tried or perhaps no conclusion can be reached. Which control strategy to use depends on the nature of the problem to be solved. Backward-chaining systems are more efficient for problems where the user wants to select from possible outcomes such as in diagnosing an illness. Forward-chaining systems perform better when data must be first assembled to perform a task. Both methods have disadvantages. Forward-chaining systems generate many hypothesis not directly related to the problem while backward-chaining

approaches can become fixed in the initial hypothesis but have difficulty shifting focus when the data available do not support them. Closer to the way human reasoning works is probably a combination of the two methods. MYCIN, an interactive system to diagnose bacterial infections, uses IF-THEN rules and provides the expert's certainty level for each rule.[2] It chains backward from a hypothesized diagnosis and uses the certainty factors of rules to estimate the certainty factors of diagnostic conclusions. If there is not enough information to narrow the hypotheses, it prompts the physician for additional data and evaluates all hypothesis. All diagnoses with high certainty values are given treatment recommendations.

THE USER INTERFACE

The user interface is really an input-output interface. It enables the user to supply information about a specific problem to the machine on his terms. This information can be supplied at run time or by routines to look for data from external sources. It also enables the user to pose queries, answer questions from the system and receive replies in a way he can understand. Because much of the information in a

---

[2] Bruce G. Buchanan and Edward H. Shortliffe (eds.), Rule-Based Expert Systems: The Mycin Experiments Of The Stanford Heuristic Programming Project, Addison-Wesley, Reading, Massachusetts, 1984, p. 65, 79.

knowledge-based system is imprecise, by giving answers to questions with a certainty factor, the user has an indication of the degree of confidence that the system has in its conclusion.

Expert systems provide a way for human experts to explicate their knowledge, and their "fuzzy" reasoning is bringing them closer to the reasoning used by people. But they can't perceive significances, draw analogies, rely on common sense or learn from experience. They are suitable to areas of depth rather than breath.

How are expert systems different from conventional data processing? Expert systems are appropriate for applications that resist conventional algorithmic solutions and also problems that are now laboriously and expensively solved by human experts. Conventional data processing automates time consuming clerical functions by processing huge volumes of data algorithmically. Expert systems address tasks performed by professionals in a few minutes or hours. With expert systems, the human problem solvers rule-of-thumb heuristic knowledge is captured and used to solve problems as conventional software uses algorithms. In traditional data processing, problems are well defined and the control structure for every possibility can be flowcharted. The logic of the program finds its data and decides what to do. With expert systems, the data determines which rule applies and decides what to do next. With conventional languages,

order of instructions is crucial. When adding or deleting code to an existing program, a mistake in the location of the change can modify the entire program, yielding a wrong result. With expert systems, rules can be grouped in any order. Execution is controlled by an inference engine, not the order of statements. In other words, expert systems are driven by data, not by a sequence of rules. Since the order of the rules doesn't matter, the total separation of data from the inference mechanism eliminates the complexity of traditional procedural programming.

Today's expert systems have several common characteristics. They have only successfully been applied to an area of narrow subject domain. This is because the science of knowledge representation has not evolved to the point of incorporation of broad, general, or common-sense type knowledge. They draw on the knowledge of experts for their problem-solving or advisory capability. They do not involve the computation of numbers and quantities but the manipulation of logical connection between facts, propositions or statements. They solve problems by determining the best sequence of rules to execute, selecting relevant rules, and combining the results in an appropriate way. They are usually not programmed in conventional manners but by using propositional logic. They can deal with uncertainty, perhaps producing several possible answers starting with what they considered most likely. Many have features to

help their users, such as spelling checkers and correctors. One of their most important attributes is their explanation facility. Many have the facility to allow the user to ask Why? and How? They are able to display the chain of inferences used to cite particular rules at each decision point. This permits human experts to correct any faults built into the system and tells what uncertainties had to be used to arrive at a conclusion, giving the user greater confidence in the results. They are easily programmed for special-case reasoning. Their most important property is that their skill increases at a rate proportional to the enlargement of their knowledge base. Their knowledge is stored as independent chunks, making them simple to add and refine.

Expert systems are not suitable for every application. They are well suited to applications that have small output in relation to input, as in diagnostic problems where the list of symptoms or observations may be lengthy but the output will be a short statement of diagnosis. If the application encompasses a number of possibilities so large as to make programming by conventional means uneconomic or impossible, expert systems can focus on the smaller number of principles which determine the answer rather than all possible combinations. At their current development, expert systems can most effectively be used within a definite domain and not where there has to be an input of general knowledge. Medicine is an ideal field for expert systems.

There are many well-defined areas where understanding of the problems is very complete so that everything that is known can be included. Expert systems can be used where there is a need to handle probabilities or incompleteness of data. They are best where goals and sub-goals can be defined and search directed along the best path to achieve a given goal. A contra-indication would be where defined processes or procedures have to be applied.

An expert system can function at three different levels. An assistant helps the expert by doing routine analysis and by pointing out those portions of the work where the expertise of the human is required. An example in this category is the Dipmeter Advisor System developed by Schlumberger Ltd.[3] This expert system reads charts produced by instruments that have been lowered into an oil well which is being drilled. The expert system reads the charts and indicates which portions of the chart should be concentrated on, eliminating a tedious analysis job. When a user talks over the problem with the system until a joint decision is reached, this is referred to as a colleague type system. If the system goes down the wrong track, the user inputs information to get it on the right track. Thus, the resultant

---

[3] "What's Happening With Expert Systems," EDP Analyzer, Vol. 23, No. 12, December 1985, p. 4.

decision is a joint one. In a true expert system, the user would accept the system's advice without question.

Users of an expert system may likewise be categorized. A sophisticated user based in a large company will most likely have access to specialized hardware and custom developed expert systems. Less sophisticated users will use one of the larger shell systems with a mainframe or possibly a specialized work station. The least sophisticated user will tend to use one of the simpler shells. A shell is a term used to describe a framework for an expert system into which the user is able to input facts and rules relative to his problem.

In a limited well-defined domain, where the skills of an expert can be summarized in a few hundred rules, a computer program will perform as well as a human being, perhaps even more thoroughly. An asset of the system is that its structure is independent of the problem to be solved. It only becomes problem dependent with the incorporation of problem dependent knowledge. And, although one cannot do without inference, the method used is not responsible for the intelligent behavior of the system - the knowledge is. Its knowledge is its power. Changing the knowledge changes the system. An expert system is a perpetual student; new knowledge can always be added. The biggest problem with expert systems is how to represent the knowledge and how to acquire it.

Knowledge representations for expert systems are currently done on a case by case basis with each application optimized for its domain specific characteristics. Standardized forms for knowledge representations are needed so that new structures are not developed for each application. Major tasks for knowledge representations research are extending the concepts of object description and extending ideas of how to represent concepts such as time and ownership. Skills we apply unthinkingly have proven to be extremely complex and difficult to program. Analogies and naive physics, common non-verbal understanding, have yet to be exploited to aid in problem-solving. For technical areas that have excellent texts and documentation, automatic translation of document into knowledge bases would be a giant leap forward. To accomplish this, documents would have to be interpreted by programs, having their meanings and relationships extracted.

The spreading of expert system activity from specialized areas into more general fields will take a long time. If expert systems are to exert major influences on the development of future software, much more work needs to be done in codifying human knowledge to make it available for problem-solving processes. Knowledge representation must include deep as well as surface, qualitative as well as quantitative, approximate as well as exact and specific as well as general information.

III. THE ACQUISITION OF KNOWLEDGE

How can an expert system be acquired? There are three ways; buying an expert system development tool or shell, buying a developed system, or building your own.

With a shell you can be productive in a matter of days. Building an expert system requires capturing expert knowledge and encoding it in computer readable form. Shells solve design problems by letting the developer concentrate on the user interface and on acquiring and expressing the knowledge on which the system is based. Advisor systems are the most common systems in use. These shells are typically rule-based but there are shells available that are a hybrid of rules and frames. A great deal of knowledge can be encoded using the IF-THEN format, but an inductive system or example-based system that generates rules from examples may be preferable where there are many examples of successful decisions, but the rules behind them are not clear. The built-in knowledge representation schemes and inference engines leave the developer free to concentrate on the knowledge.

In any case there is a need for knowledge engineering. Knowledge engineering is the parallel to software engineering of conventional technology. The job of a knowledge engineer is to ferret out expertise, represent it in data structures, and put it to work using inference methods to

solve problems. The communication and problem-solving skills for knowledge engineering are closest perhaps to today's systems analysts. Knowledge must undergo a transformation before acquiring commercial value. The knowledge engineer selects an appropriate scheme for representing knowledge; frames, rules, etc., to transform the know-how into a knowledge base to be used by an inference engine. The process of refining knowledge continues until system performance is adequate.

Expertise in a subject area is comprised of vocabulary definitions, objects and their relationships, rules, constraints, hypothesis, heuristics, description of processes and experience. One of knowledge engineering problems is tying together all of this information. The expert system itself cannot turn unstructured heaps of valid rules into effective problem-solvers. The expert system is not a human who learns easily. It must be spoon-fed, and even the simplest problem brings a vast amount of knowledge to bear to solve it.

There are basic steps in the knowledge engineering process. The engineer and expert must first work together to identify the problem area and define the scope and objectives of the expert system. They must define the key concepts, relationships and information flow describing the problem solving process. The knowledge engineer must map the key concepts into a formal representation and formulate

rules. Both the knowledge engineer and the expert must work together evaluating the performance of the prototype program. For the ACE system (Automated Cable Expertise) the domain was cable analysis.[4] Knowledge engineers interviewed cable analyzers from different operating telephone companies to learn how they analyze outside problems. ACE's expertise is a compilation of the expertise and strategies of several cable analysts.

The acquisition of knowledge for PRIDE (Pinch Roll Transport Interactive Design Environment Expert), an expert system for the design of paper handling systems, was accomplished in the following manner.[5] Design experts were asked to collect a set of representative design cases and asked to perform detailed design on the selected set. Outlines of the design process of the designers were sketched by the knowledge engineer. The engineer then created a document describing how experts designed paper transports and circulated it to the experts for feedback and revisions. The experts were then asked to design more paper transports so that their behavior could be compared to the document. This highlighted mistakes and ambiguities contained in the docu-

---

[4] F. D. Miller, J. R. Rowland, and E. M. Siegfried, "ACE: An Expert System For Preventative Maintenance Operations," Record, January 1986, p. 23.

[5] S. Mittal, C. L. Dym, and M. Morjaria, "PRIDE: An Expert System For The Design Of Paper Handling Systems," Computer, Vol. 19, No. 7, July 1986, pp. 102 - 114.

ment. Finally, someone who was not an expert was asked to use the document to design a paper transport for a new set of requirements. Their failure pointed out that the experts had not been questioned hard enough to understand their reasoning process behind their decisions. Their common-sense understanding had not been transferred during the acquisition process.

The developers of MYCIN experimented with computer-based tools to acquire knowledge from experts through interactive dialogues.[6] Communication between experts and knowledge engineers was slow, and it was desired to speed up transfer of expertise. The first tools included a rule language that allowed entering a new rule in quasi-English. MYCIN then translated the rule into its internal LISP representation and translated it back into English to point out a version of the rule as it understood it with the user being asked to approve or modify it.

There are problems associated with each of the prior mentioned forms of knowledge acquisition; an expert interacting with a knowledge engineer and an expert conversing with an intelligent editing program. The former has commu-

---

[6] Bruce G. Buchanan and Edward H. Shortliffe (eds.), Rule-Based Expert Systems: The Mycin Experiments Of The Stanford Heuristic Programming Project, Addison-Wesley, Reading, Massachusetts, 1984, pp. 153 - 154.

nication problems from the expert to the knowledge engineer, the latter from the expert to the program.

With traditional programming, all possible outcomes are considered beforehand. This isn't possible with an expert system. Even when a general approach can be formulated, a large amount of exceptions that an expert has forgotten to include are bound to appear. The expert might not have an explanation for some of his decisions; human reasoning happens on many levels and is not well understood. There are no standards for representing knowledge and performing inference. Knowledge engineers work by trail and error. The knowledge engineer must form an initial model of the experts reasoning processes without a total understanding of the knowledge. As the expert system model is subjected to tests comparing its reasoning to the expert, knowledge is added incrementally as the process is iterated. Expert systems are designed to allow changes, but it is important that the knowledge engineer formulate a proper model of the experts domain to structure the knowledge base efficiently. If the wrong problem-solving approach has been chosen, it may be necessary to restructure the entire knowledge base. Choosing an entirely different form for knowledge representation, such as frames instead of rules, or choosing a different inference strategy, such as backward chaining to forward chaining is akin to starting over completely. There are no software tools for doing such a conversion.

Enormous time delays can result when information has been omitted. Discovering omissions at later stages can be expensive if the knowledge engineer has moved onto other projects and needs to return to reprogram the system with additional information.

But the main bottleneck of expert system development is rationalization of the intuitive process. How an expert articulates the process he uses to arrive at a conclusion is very often only the tip of the iceberg. There is a great inability to express subconscious knowledge. Experts find it easier to practice their skill than explain it. At this stage of development, knowledge is obtained one-on-one between the knowledge engineer and the expert. Estimates of rule acquisition rate are a painstaking one rule per hour.

MYCIN has been witnessed to out-perform acknowledged medical experts.[7] This is not surprising since the combined experience of many experts might be expected to be superior to any one. The need to combine the knowledge of multiple experts is inherent in expert systems. But this creates a need for a way to check consistency of information and possible conflicts of information from the experts. There must be one special acknowledged expert designated on the

---

[7] Frederick Hayes-Roth, Donald A. Waterman, and Douglas B. Lenat, Building Expert Systems, Addison-Wesley, Reading, Massachusetts, 1983, Vol. 1, p. 10.

project team capable of resolving the above issue. Knowledge engineers are limited in their ability to maintain consistency and resolve conflicts. One expert must maintain control to ensure the quality of the knowledge base.

A distinction needs to be made between the expert contributing to the knowledge base and the end user. Who should be entrusted with the task of modifying the knowledge base? The real value of the system is its knowledge. Incompetent information within the system can grow if casual users are privy to this access. Different users having different levels of system security is perhaps one alternative.

Without a thorough command of an application, the knowledge engineer is unlikely to spot omissions until the program fails to work. Without an understanding of expert system programming, the expert might overlook the importance of certain data. Using both the knowledge engineer and experts to set up the system is inviting problems, creating scheduling delays and increasing cost. An approach that would make knowledge engineers unnecessary is to design intelligent systems that can prompt experts to program the applications without the intervention of a knowledge engineer. This could be in the form of answering questions drawn from the experts domain of knowledge. The result is an expert system that can be designed and maintained by the same experts who will use it and avoiding the expense of a

knowledge engineer. TEIRESIAS is the interactive rule acquisition mechanism of the MYCIN system. It provides capabilities of both explanation and acquisition functions and facilitates the transfer of expertise from the physician to the MYCIN system.[8]

The last method of knowledge acquisition is <u>automatic skill acquisition.</u> This would require no expert but only a large data set from which to learn heuristic rules, for example text books to a text understanding program. This function is not yet available. But when it is, expert systems that are capable of acquiring skills automatically will be able to evolve to experts.

One of the attributes of an expert is the large amount of knowledge he has in his field. The advent of expert systems could herald an end to the era where people study books or train with an expert for years before practicing his craft. Improved approaches to knowledge base management should be explored since this will affect the acceptance or rejection of such systems by their intended audience. Allowing the expert to build his own knowledge base without an AI software specialist could not only allow the best translation of the experts knowledge but could also resolve

---

[8] Bruce G. Buchanan and Edward H. Shortliffe (eds.), <u>Rule-Based Expert Systems: The Mycin Experiments Of The Stanford Heuristic Programming Project</u>, Addison-Wesley, Reading, Massachusetts, 1984, pp. 171 - 172.

the acute shortage of AI experienced personnel that could constrain market development.

IV. TECHNICAL ISSUES - CURRENT AND FUTURE

To popularize expert systems, most development tool vendors believe the technology must be available in today's popular languages such as C, PL/1 and COBOL and run on computers other than LISP machines. In that way, expert systems can be tied to new and existing applications. However while some proponents of expert systems agree that they must move away from LISP, others present valid arguments for staying with LISP.

One reason that LISP is not a good commercial tool is that it suffers performance problems at run time because of the overhead it generates. Because list processing is the heart and soul of LISP, normal limits by most languages on the size of data structures are not acceptable in LISP. LISP requires an enormous amount of memory so that personal computers are generally not useful for full-scale LISP applications. Another problem lies in the fact that LISP is extendible; functions that are defined become part of the environment and can be used to define other functions. Because of this there are many dialects of LISP, and portable LISP software is virtually nonexistent. Thus, although LISP is an excellent development environment with great power and flexibility, it has several disadvantages.

The major difference between dedicated LISP machines and computers that run LISP is that the dedicated machines

have been embedded with more than 12,000 LISP functions in the programming environment to optimize the language.' Proponents of LISP point out that there are good versions of LISP available on a variety of mainframes, and that LISP compilers are getting more efficient and hardware speed so fast that the run-time performance issue will become mute. They argue that transition of expert systems from the research lab to the field will be smoother if the same language is used. They claim that translating an application from LISP to another language is not justified if the LISP version is fast enough, small enough to fit on the computer used and can be integrated with the commercial environment. Run-time LISP systems including only the functions needed are claimed to be small enough.

There seems no doubt that the language of choice for research and development is LISP. But those who use tools to build expert systems are developing applications and their language of choice is the tool they use to build the application. Traditional languages have less power and flexibility but are optimized to run efficiently. And most existing software is written in traditional programming languages. And so while some computer vendors have been introducing LISP compilers in their machines as a step in

---

' G. Rifkin, "Toward The Fifth Generation," Computerworld, Vol. 19, No. 18, May 6, 1985, p. 20.

being able to support AI programs, many expert system tool vendors are introducing new versions of their products in C language.

Another issue is the choice of knowledge and inference representation. The IF-THEN format represents the most common way of expressing knowledge. With effort, a developer can code a great deal of knowledge based on IF-THEN rules. Frames can describe how facts are related to each other, but systems built around frames can be more difficult to program and debug. The order of chaining may also be significant if the inference engine considers its job done when it finds a single valid conclusion.

As technology is applied to larger and more complex problems, the size and slow speed of execution on conventional machines might doom such attempts. The R1/XCON program designed to configure Digital Equipment Corporation's VAX computers is such an example.[10] R1/XCON contains approximately 1500 rules and several hundred facts describing a partially configured VAX. Running on a DEC VAX 11/780 computer and implemented in OPS5 language, configuring an entire VAX system requires hours of computing time on

---

[10] S. J. Stolfo, "Knowledge Engineering : Theory and Practice," pp. 101 - 102. Proceedings of Trends and Applications 1983. Automating Intelligent Behavior. Applications and Frontiers, May 25 - 26, 1983, Gaithersburg, Maryland. IEEE : New York, 1983.

a large expensive computer. Performance will worsen as systems are designed with tens of thousands of rules. Meta-rules, or rules about rules, could embody strategies for selecting paths of reasoning. Third-order rules could be used to select from or order the meta-rules. This kind of strategy information may translate directly into increased speed, since fewer rules need to be tried, or to no degradation in speed even with large increases in the number of rules.

But computer scientists suspect that the sequential approach of computer processing may by inadequate for building computers able to process information fast enough. Today, even with programs that use a few hundred facts they take too long to process. The demand for increased speed can be met by a radical change in computer architecture, from a single serial processor to a computer which is an aggregation of many parallel processors that perform independent operations concurrently. Parallel processing is different from serial processing in that rather than having a single serial processor performing operations sequentially, the machine will employ thousands or millions of processors simultaneously on a single task or multitude of tasks. However the problem of programming for parallel systems is a major obstacle yet to be solved.

The prospect for large expert systems will be enhanced when ways are found of reducing processing time. Extensive

searching of a knowledge base is required to find matches for rules and facts. The time to do this sequentially can be a constraint. A parallel processing machine which coordinates processors working on the same data and sharing the data between them without affecting data integrity, will give the speed of response that is so important for user acceptance.

V. MIS STAFF - FROM DATA MANAGEMENT TO KNOWLEDGE MANAGEMENT

Even though most computer-using organizations have been installing and using new application systems for twenty to thirty years, employees still view new systems with some fear and resistance. Likewise, certain topics make data processing people uneasy. One of those is Artificial Intelligence. This is partly because of unfamiliarity, as until recently it was mainly the subject of research.

Upper management may well see expert systems as addressing their decision-making needs. They might welcome software that can capture expertise, think like an expert and made decisions; in other words help them with tasks that are traditionally theirs. They could be enthusiastic when they understand the advantages of expert systems. Usually upper management is mainly concerned with the bottom line - cost, changes in staff performance and changes in market share.

Middle management and professionals may feel their job knowledge will be transferred to the computer and they will no longer be needed. Others will feel that the interesting and challenging parts of their jobs will be put on the system and will be left with the dull work. Still others may feel that their company will find out how trivial their job is. If so, they may become the main opposition to the use of expert systems unless it can be shown to work to

their advantage. But other managers want to get involved with expert systems. They may be motivated to solve a problem that cannot be solved with conventional approaches. Maybe they want to be able to discuss the topic with their management or perhaps they perceive that levels of employment will be unaffected and performance improved. They might want to understand their role in the technology.

Expert system technology has been held back primarily because it has required programmers and knowledge engineers with experience in AI and LISP programming coupled with computers that run the language efficiently. This acute shortage of AI-experienced personnel can be expected to place a significant constraint on market development for the next five years.

Yet corporate management, in undertaking expert system projects, establishes special AI task forces, hires specialized programmers and purchases computer systems, creating waves within the well-established MIS group. They think they need a fresh point of view. But an expert system is still a system-developed project and the separation of expert system development teams from application programmers is unnecessary. Rather than needing new people, the manager needs new skills. Instead of taking specialized programmers and interfacing them with the problem, he needs to take the people with the problem and teach them how to program. Rather than restructuring data management departments,

managers should develop qualified knowledge engineers and designers with cost-effective in-house training. To be conversant with the technology, training is necessary in AI concepts, expert system development, using expert systems to make business decisions, and understanding system possibilities and limitations.

Corporations exploring expert systems should start small and demonstrate the viability of expert systems as staff support, showing potential benefits. Selecting something simple for a first application is almost sure to turn out to be more complex than was first believed. A colleague type system can show employees that they will still have their jobs and that it will only help with routine aspects. A key employee approaching retirement creates the perfect vehicle for training other people in the job as well as helping them to do the job. In an area where there are only a few experts under severe job pressure, receiving urgent phone calls at night and on weekends, employees would welcome a system to remove some of the pressure. The Travelers Corporation in Hartford, Connecticut has developed and expert system for diagnosing problems with IBM 8100 computers.[11] They have a nationwide network of 8100's serving 10,000 terminals and processors. The incentive to develop

---

[11] "What's Happening With Expert Systems," EDP Analyzer, Vol. 23, No. 12, December 1985, p. 5.

the system was job pressure on the company's diagnostic experts for the 8100. The system was created in three months using the M.1 shell from Teknowledge, Inc. Now a help desk staff member can handle a trouble call, ask questions, enter the answers, diagnose the problem, and give instructions. Downtime of 8100's has been reduced and so has the pressure on 8100 experts. The ACE system (Automated Cable Expertise) an expert system that performs preventive maintenance operations by analyzing thousands of customer trouble reports for signal problems, was also well accepted by local telephone company's maintenance center personnel for the above-mentioned reason.[12] Other bridging techniques could be to automate a complex job that employees have a hard time doing accurately (this has been done with equipment configuration checking), or a decision area that is important but not performed frequently enough to develop expertise.

How should the information systems function be organized to properly support system function? The MIS department could function as consultants for expert system tool selection, training users in applications, as well as developing maintaining and supporting custom systems. MIS managers will become knowledge managers in charge of corporate

---

[12] F. D. Miller, J. R. Rowland, and E. M. Siegfried, "ACE: An Expert System For Preventative Maintenance Operations," Record, January 1986, pp. 20 - 25.

knowledge. Taking the abilities of the best people and making them available to other people upgrades everyone's capabilities. People who are very good get to move on to even harder problems. A system incorporating the expertise of a retiring programmer could train his replacement not only in the ways of data processing tools but also in the ways they are used at a particular site. Managers and their staff could use a system that stores expertise about company policies. MIS staff can be freed from the clerical coding tasks and frustrating testing of fourth generation non procedural languages which require specification of the sequence of steps necessary to achieve the desired result. This makes for out of place statements, erroneous logic, multiple logic paths and if-then-else constructs leading to bugs and the requirement to test, re-code, retest, etc.

In taking MIS managers from data managers to knowledge managers, expert systems will require a realignment of the data processing professional. The data processing managers that had difficulty adapting to the concept of management information systems will have a harder time adapting to expert systems. The key to expert system acceptance is <u>training.</u> Once a manager realizes their help in decision-making and preservation and dissemination of expertise, there will be no looking back.

VI. EXPERT SYSTEM EVALUATION

Evaluation is an important aspect of expert system development. It enables a feedback process to take place whereby comments and critiques serve as a basis for iterative refinements to the system. Why is it necessary to evaluate an expert system? Evaluations measure the software accuracy and usefulness. Even if an expert system is delivered on time, within budget, and performs its functions efficiently, a user might still be unhappy with a design that is hard to understand and modify or difficult to use. Basically, evaluations facilitate convincing the end users of system viability and spur system performance improvals by spotting deficiencies.

Expert systems need to explain the line of reasoning that led to problem solutions. This helps refine and improve the system by clarifying steps that led to an incorrect answer. If an assistant system is unable to explain its line of reasoning, it will not gain the initial confidence of the users who have to take responsibility for acting on its recommendations. There is an element of legal liability in this. For example, who is responsible for an incorrect medical diagnosis - the expert system developers or the person using its advice.

It is necessary to provide a means by which the system can explain its decisions because expert systems are applied

to areas in which computer assistance is uncommon. They claim to be capable of performing tasks previously requiring the intelligence of a human expert. An expert system must be accepted by users to be successful. Being able to explain why a question is being asked by the system and how it arrived at its conclusion will go a long way in assuring users that its reasoning is logical and its advice appropriate. TEIRESIAS contains the knowledge base revision function of the MYCIN system.[13] It suggests what kind of rule will correct the problem and offers to write a specific form of the rule if it can. It then checks with the user to see if that specific rule makes sense and offers the user an opportunity to change it. If the user edits the rule, TEIRESIAS matches the new rule for consistency and correctness of syntax. As a check, TEIRESIAS reruns the current case with the new rules.

Once a prototype has been built, the task of evaluation should begin. Examination of program reasoning when errors arise can point out areas where the knowledge is weak. When the system is critiqued by the real expert, many special case and exceptions in the way the experts apply knowledge are bound to become apparent. But the system should also be reviewed by non-expert users who will determine whether or

---

[13] Frederick Hayes-Roth, Donald A. Waterman, and Douglas B. Lenat, Building Expert Systems, Addison-Wesley, Reading, Massachusetts, 1983, Vol. 1, p. 152.

not the information requested and returned is understandable and helpful. If the rules of the system are impossible for him to read, debugging and validating are more difficult than they should be. Meta-rules, rules about other rules, can be used to justify rules and thus enhance the systems explanation abilities.

What criteria should be used for expert system evaluation? Is the knowledge representation scheme adequate? Does the system come up with right answers for the right reasons? Is its knowledge consistent with the experts? Is it easy for users to interact with the system in terms of the choice of words used in questions or responses? Other criteria could include its portability. Can it be executed on hardware configurations other than its current one? Is the output of the system consistent and uniform with regard to its notation? Can the system be easily modified?

How can a standard be defined for expert system correctness? Is an expert system to be judged correct if it produces the correct answer to a problem or if it produces that same answer as a human expert would when presented with the same information? It is reasonable to suggest that an expert system performs at expert level and was in fact correct if it agreed with the expert. It is interesting to note that MYCIN's evaluation by expert physicians when they knew they were examining the output of a computer program

reflected their bias regarding computer systems and led to a blinded study design.[14]

In the evaluation of an expert system prototype READ (Requirements Engineering Automated Development) which was developed for use of determining software functional requirements for command management activities of NASA supported satellites, the following evaluation criteria were used: ability to update, ease of use, hardware, cost-effectiveness, input-output content, quality of decisions and design time.[15] Three users of the READ system compared the system to using a human expert alone. Based on the seven criteria, all three rated the system superior to a human expert.

Application of graphics techniques to knowledge-based management tools can aid in visualizing the interaction of various components in the knowledge base or anticipating the consequences of these interactions. This is especially important when the end user is not the same person as the expert. ONCOCIN is a medical consultation system designed to help physicians with the management of cancer patients

---

[14] Frederick Hayes-Roth, Donald A. Waterman, and Douglas B. Lenat, Building Expert Systems, Addison-Wesley, Reading, Massachusetts, 1983, Vol. 1, pp. 263 - 264.

[15] J. Liebowitz, "Useful Approach For Evaluating Expert Systems," Expert Systems, Vol. 3, No. 2, April 1986, pp. 86 - 96.

enrolled in experimental chemotherapy.[16] Early version of ONCOCIN had no facility for browsing the knowledge base. Knowledge engineers relied on a 150 page listing and a booklet of chemotherapy cards used by oncologists when caring for patients. A browsing system has been developed for ONCOCIN that provides an overview of the major data types represented as hierarchies. A graph represents the relationships between diseases and chemotherapies. Information about the diseases or chemotherapies on the graph can be retrieved using a mouse pointing device. Using the mouse causes a special menu to appear on the screen which is used to select specific areas of interest. The chemotherapy cards can also be both graphically and textually displayed.

Graphics can enhance textual explanations with the diagram giving a comprehensive view of system components as well as detailed descriptions of individual facts. A carefully constructed graph or diagram can convey a great deal of information in a form which is not overwhelming or confusing, and that would be impossible to display in text. ONCOCIN's users have control over the amount of detail to be displayed and the form of that display. The same information can be described with text or represented in a graph or

---

[16] S. Tsuji and E. H. Shortliffe, "Graphical Access To Medical Expert Systems: I. Design Of A Knowledge Engineer's Interface," Methods Of Information In Medicine, Vol. 25, No. 2, April 1986, pp. 62 - 70.

diagram. This provides a comprehensive overview that is hard to obtain from written documentation.

The ultimate criteria for expert system success is whether an expert system is actually used for expert consultation. A key ingredient in this success is to involve the eventual users in the evaluation of the system while building it.

VII. EXPERT SYSTEMS - WORTH THE COST?

Expert systems are the focus of many efforts to commercialize Artifical Intelligence technology for the computing mainstream of MIS. Expert systems constitute by far the bulk of activity in the commercialization of Artificial Intelligence. The Gartner Group Inc., a Stamford Connecticut market research firm forecasts that by 1990 the market could jump to $275 million for expert system tools, and $350 million for expert systems.[17] By 1990, MacKintosh International forecasts the expert system market for France, Germany, Japan, UK, and USA to be almost $1 billion.[18] In the past, the major sources of expert systems cost were special purpose processors that had to be developed. But now expert systems are being built to run under conventional operating systems on conventional hardware. They can run on a firm's normally installed machines. Very large companies with a fixed policy for purchasing equipment would demand that compatibility. Yet little AI development is funded from corporate budgets, implying that the average company does not believe in the commercial opportunities of AI technolo-

---

[17] T. Manuel, "What's Holding Back Expert Systems?" Electronics, August 7, 1986, pp. 60.

[18] Mackintosh International, Expert Systems 1985-1990, A Report prepared by Mackintosh International Limited, 1985, Vol. II, p. 82.

gy, or will not launch an internal AI shop without a clearly visible and feasible cost-effective product in mind.

Corporate expert system funding can take various shapes and can be done at different costs, depending on company size, available financial and human resources, and general commitment to high technology. The choices range from expert system shells to custom expert system development.

Expert system shells come in a wide variety of sizes and types. These range from $50 personal computer based packages through mainframe development systems with price tags in tens of thousands of dollars.[19] An expert system shell can be a relatively inexpensive way to verify the suitability of a task to expert systems. Another advantage of purchasing a shell instead of writing a system from scratch is that the knowledge representation schemes and inference engines are built into the shell, leaving the programmer free to concentrate on the best way to express the expert knowledge.

From a cost-estimate point of view, custom expert system development resembles typical custom software development, but custom expert systems are much more expensive. This is due to the scope and depth of the project and the

---

[19] M. Williamson, "Expert System Shells - Design Tools Help MIS Answer Management's Call," Computerworld, Vol. 20, No. 28, July 14, 1986, p. 52.

wages of the people involved. An organization would turn to expert systems to solve a problem that other computer science techniques have failed to solve. This would typically be of significant size and depth. Because of the scarcity of AI expertise in the market, knowledge engineers and other expert system professionals command a high price. Good LISP programmers earn $40,000 - $60,000, with consultants costing even more.[20] However, consultants give access to top personnel and require no capital or long-term commitment. Based on Arthur D. Little, Inc.'s experience developing more than thirty large-scale knowledge-based systems for Fortune 500 companies, an expert system development project costs from $150,000 - $200,000 for each person-year of effort.[21] This takes into account wages, benefits, hardware and software support, administrative costs, and travel. The entire project requires 10 - 16 person-years of effort, costing $1.5 - $3.2 million.

The project can be divided into three stages.[22] The proof-of-concept stage is used to develop enough essential components to determine whether AI techniques can solve the problem at hand. This includes the building of a small

---

[20] J. R. Davis, "Custom-Developed Expert Systems Offer Strategic But Costly Alternative," Computerworld, Vol. 20, No. 2, January 13, 1986, p. 52.

[21] Ibid.

[22] Ibid., pp. 52 - 53.

knowledge base, skeletal control logic and user interface at a level of effort of 1 - 2 person-years and cost of $150,000 - $400,000. The demonstration stage, with a beefed up knowledge base, can convince top management to invest in the full blown system. The level of effort and cost can be expected to be the same as the proof-of-concept stage. The prototype stage, a full working model with a complete knowledge base, control logic, and user interface, lasts 8 to 10 person-years at a cost of $1.2 - $2.4 million. This is easy to believe, considering the average cost of formulating and implementing rules based on the number of person-hours spent in construction versus the number of rules specified. For example, CLOT, a consultant for bleeding disorders, required approximately sixty hours to specify sixty rules, working out to one person-hour per rule.[23] Even after a total resource commitment, thorough validation, debugging, and complete documentation, raises the level of effort to 10 - 16 person-years and $1.5 - $3.2 million.

This creates a need for information systems executives to quantify the value of such a system for their corporate environment. In order to determine how much might be saved by distributing, supplementing, or replacing expertise with

---

[23] Bruce G. Buchanan and Edward H. Shortliffe (eds.), Rule-Based Expert Systems: The Mycin Experiments Of The Stanford Heuristic Programming Project, Addison-Wesley, Reading, Massachusetts, 1984, p. 326.

software, an understanding must be reached as to how much an expert costs.

The annual cost of an expert with an annual salary of $54,000 is $122,569. This assumes employment by a company of at least five hundred employees, and having twenty years of experience.[24] This takes into account salary, benefits, recruitment cost, replacement cost, profit sharing, travel, required facilities, administrative support, and management and supervision of the expert.

If the expert system is viewed as a replacement for human beings, the value of the system would equal the annual cost of the employee multiplied by the expected life of the system. At this point in the evolution of expert systems, that is not realistic. However, expert systems can be profitably used in a number of ways.

It can be used to retain the expertise of departing employees. The value of the system would equal the cost of the departing employee multiplied by the number of years the system would be used. If the employee was replaced, the pay differential of the two employees should be considered. Perhaps having a replacement could be delayed because of the presence of the system. In this case the cost of the poten-

---

[24] L. Allen, "The Cost Of An Expert," Computerworld, Vol. 20, No.29, July 21, 1986, pp. 62 - 64.

tial employee and the number of months hiring was delayed could be used. The cost of bringing in a consultant, training a replacement, and the cost to solve problems without the system are all considerations.

Also, it is far less expensive to ship software around the country than people, and with his knowledge operating independently, the expert could be freed up to concentrate on more development. Very often senior people are recruited to train novices, in which case the value of the system could be found from the cost of the senior per hour, the average number of hours each novice requires, and the number of novices. Another value, harder yet to quantify, is that of ascertaining the effect of rule changes of company policy or procedure; for example clients gained or revenue generated.

Program maintenance should be another consideration. With procedural programming, it often takes the form of restructure and retrofit rather than extend and change like expert systems. Each modification must be retested; and often changing one part of the system unexpectedly changes another apparently unrelated part. Maintenance consumes more than two-thirds of programming effort during the life

of a conventional system.[25] Saving in this area could potentially be more rewarding than saving in the initial development.

The director of Artificial Intelligence at Westinghouse claimed to increase throughput in one factory to the extent of increasing business volume by more than $10 million a year by using Expert Ease, an expert system development tool.[26] Texas Instruments used a system developed on its Personal Consultant tool for maintaining part of a semiconductor wafer manufacturing line. This system has approximately 450 rules, took 4 to 5 person months to develop, and is claimed to have produced a 10% increase in production output.[27]

There are three ways to rate potential expert system applications.[28]

1. Fact/Rule ratio
2. Decision tree differential
3. Return on investment

---

[25] R. Philips, "Can Fifth-Generation Software Replace Fallible Programmers?' Computerworld, Vol. 18, No. 29, July 16, 1986, p. 28.

[26] "What's Happening With Expert Systems," EDP Analyzer, Vol. 23, No. 12, December 1985, p. 5.

[27] Ibid.

[28] L. R. Harris, "When Is An Expert System Right?" Information center, Vol. 2, No. 7, July 1986, pp. 96 - 97, 119.

The first two items assess the complexity of applications while the third is important because the financial payback for solving each application is unrelated to technical complexity. More complex problems have a higher proportion of rules; simpler problems have a higher proportion of facts. Fact-intensive problems put less pressure on the inference process and the fact base is easier to build and maintain because facts, unlike rules, are always true. Therefore, the best application to select is one with the fewest rules and facts and the highest fact/rule ratio.

Another way to assess complexity is to view complexity in terms of a decision tree. The degree to which a decision tree was successful indicates how successful an expert system is likely to be. The smaller the differential or measure of the degree to which a particular application is beyond decision trees, the more solvable the application.

Return on investment is the ratio of the value of solving the problem to the cost of building the system. Obviously, the applications to choose are those with the highest paybacks that are easiest to implement.

Technology is often accessed from an economic perspective. So it is no surprise that as corporations turn to technology to improve their competitive edge, the value of a company's most precious resource, its expertise, is being weighed against the cost of developing and maintaining an

expert system. The best way to justify a new technology is to focus on its ability to generate additional revenue. Quantifying the value of existing corporate investments in expertise, and reviewing the effects of enhancing this expertise by expert system technology is one alternative.

VIII. INTEGRATION WITH TRADITIONAL SOFTWARE - A MUST

Integration of expert systems with mainline computing is the key to successful applications based on expert system technology. Data base technology has progressed rapidly in recent years. There are commercially viable Data Base Management Systems that use hierarchial, network and relational data models to store and retrieve large quantities of data on everything from microcomputers to mainframe. Meanwhile, Artificial Intelligence technology has made expert systems more useful and practical. But the two areas have been developed independently. Existing expert systems lack a Data Base Management System's ability to efficiently search and exploit megabytes of data, and Data Base Management Systems were never designed to cope with the kind of rule-based information found in expert systems. Realistically however, given the large application base representing billions of dollars of corporate investment, FORTRAN and COBOL are here to stay. Rewriting all code in these languages is an impossibility. Since one attribute of an expert system is access to a large amount of knowledge, why not exploit the considerable knowledge already captured in the corporate environment?

Computer manufacturers are finding out from their experiences that to become successful, expert systems must be integrated into other applications. This requirement is paramount to their customers that purchase expert system

applications and tools. To popularize expert systems most development tool vendors believe the technology must be available in today's popular languages such as C, PL/1 and COBOL. By running on computers other than LISP machines, expert systems can be tied to new and existing applications to enhance their capabilities. Migration of expert system tools away from the LISP language and LISP machines signals an acceleration of expert system applications towards MIS.

There is no doubt that there exists considerable market opportunity for applying expert systems to existing applications. Is it better to adapt expert systems to an existing data base environment or to exact a transformation of existing data bases for use by expert systems? Most evidence points to the former. Problems associated with programming effort required for reliable data transfer, testing of system function and integrity in transformation of a hierarchial data base to an expert system, for example, currently seem insurmountable. Expert systems however, can be adapted to run on any type of computer, be it micro, mini or mainframe and can be buried within or sit atop conventional products. For example, a credit authorization package could incorporate an expert system to handle marginal cases now referred to human operators for resolution. In the MYCIN system the number of questions put to physicians could have

been significantly decreased by lookup in automated library records.[29]

As expert system technology is merged into existing corporate data processing environments, users could invoke it from the mainframe-based system via a simple subroutine call. Likewise expert systems often need to execute conventional data processing applications as a subtask. An expert system could sit atop preexisting systems. A case in point is an order processing system. This type of system is generally very complex, requiring frequent and expensive maintenance. Allowing an online or even batch order processing system to call an inference engine when required and provide the inference engine with direct access to the data base management system is in fact imbedding the expert system within the application. Many other expert system components such as an order hold expert containing all rules for putting orders on hold, an inventory allocation expert, containing all rules to allocate inventory, or even a carrier selection expert for optimum transportation selection could be buried within an application; creating a structure of programs atop a host of expert components. Separate

---

[29] Bruce G. Buchanan and Edward H. Shortliffe (eds.), Rule-Based Expert Systems: The Mycin Experiments Of The Stanford Heuristic Programming Project, Addison-Wesley, Reading, Massachusetts, 1984, p. 692.

interfaces can simplify access to data bases without requiring modifications to them.

The Ace System (Automated Cable Expertise), an A T & T product for cable analysis is a proven example of this process.[30] It is an addition to the ARSB (Automated Repair Service Bureau) a networked set of systems that telephone companies use to maintain loop cable plants. ARSB has two data bases which guide local preventative maintenance routines. But even with those data bases, thorough consistent analysis wasn't always achieved because the supply of qualified cable analyzers was limited. ACE was proposed as an assistant to the cable analyzer. Maintaining a dialogue between the ACE system and the ARSB system is facilitated by REX (Remote Execution) a special communications module, creating a link between an ACE analysis that needs data and the ARSB.

Expert system technology is not a stand alone discipline. Without access to corporate data base management systems the potential for data duplication is high. Recognition of the fact that many experts can be found even in the most typical of data processing applications and exploiting that fact to make effective use of powerful

---

[30] F. D. Miller, J. R. Rowland, and E. M. Siegfried, "ACE: An Expert System For Preventative Maintenance Operations," Record, January 1986, pp. 20 - 25.

complex programs already existing is necessary. An astounding potential exists for knitting together diverse sources of information residing in different data bases and reflecting different formats and coding practices, to produce an integrated interpretation in a way perhaps never done. Part of the key strategy to widespread use of expert systems will be their integration into conventional software, not instead of conventional software.

IX. CLOSING COMMENTS

The development of knowledge-based expert system technology is an important achievement of Artificial Intelligence research. But a number of practical problems need to be solved if the field is to fulfill its promise.

There is an element of sensationalism and intrigue about expert systems. Critics claim that existing expert systems have no more connection to intelligence than conventional computer programs. Then there are the philosophical issues of machine simulating thinking. Can the associations and generalizations that language evokes in people and that constitute the essence of meaning and thought be duplicated by machine?

Although expert system technology originated in the research facilities of universities, I believe they will be taken for granted as a basic part of computer software technology in the future. Easy access to expert advise and knowledge will have a revolutionary impact when some of the present roadblocks are removed.

Present applications of expert systems are in narrow, specific problem areas because of the difficulty of representing knowledge, especially common-sense knowledge. As this changes, expert systems will change to possess broader knowledge. Successful development of automatic and interactive knowledge acquisition tools will reduce the knowledge

engineering problem of acquisition, the other major obstacle to expert systems. Creating natural-language computer interfaces is the first step in the complicated process of getting a computer to comprehend syntax and semantics.

The future of the expert system market depends largely on the type of software available to users. Higher level languages and better shells will spur user adoption of expert systems. Building an effective knowledge base for applications relevant to the data processing industry requires a large investment of resources. In order for the broad penetration of expert systems into industrial and commercial applications, organizations must perceive their high value and attractiveness.

Closer integration of knowledge systems and data processing is crucial. Intelligent facilities to automate the building of expert systems for the transfer of this technology is one possibility. But more likely, it will become commonplace to embed expert systems within corporate applications. This will push the two technologies together to the point where expert systems are no longer separate identities. In order to achieve this, the programmers of business and industry must be taught knowledge engineering. Resistance to this "New Wave" thinking should be dispelled by realization of what can be expected from such systems. There can be no argument against improved productivity and service and better levels of judgement. In a way, a kind of

immortality can be achieved by the preservation of one's knowledge. This is not to advocate replacement of human workers with machines or an attempt to put a price tag on human beings.

One of the great differences between humans and computers is their memory organization. While computers store information in discrete physical locations, the brain distributes information storage. The pattern of information distribution may relate to the meaning of the information. A computer hacks away at data, testing for patterns that are obvious to humans. When the secret of the process of human reasoning unfolds to us, today's expert systems will seem Neanderthal by comparison.

X. BIBLIOGRAPHY

Allen, L., "The Cost Of An Expert," Computerworld, Vol. 20, No.29, July 21, 1986, pp. 59 - 68.

Buchanan, Bruce G. and Edward H. Shortliffe (eds.), Rule-Based Expert Systems: The Mycin Experiments Of The Stanford Heuristic Programming Project, Addison-Wesley, Reading, Massachusetts, 1984.

Clare, J., and N. Ostler, "Fifth Generation: Long Road To Intelligent Systems," Scicon Software And Service Review, No. 3, 1984, pp. 16 - 19.

Cox, I. J., "Expert Systems - Programs That Behave Intelligently," Electronic Systems News, Autumn 1985, pp. 6 - 8.

Davis, D. B., "Artificial Intelligence Enters The Mainstream," High Technology, July 1986, pp. 16 - 21.

Davis, J. R., "Custom-Developed Expert Systems Offer Strategic But Costly Alternative," Computerworld, Vol. 20, No. 2, January 13, 1986, pp. 52 - 53.

Durham, T., "Fifth Generation Fever," Practical Computing, Vol. 7, No. 10, October 1984, pp. 115 - 117.

Eddy, W. F., and G. P. Pei, "Structures Of Rule-Based Belief Functions," IBM J. Res. And Dev., Vol. 30, No. 1, January 1986, pp. 93 - 101.

Feigenbaum, E. A., "Artificial Intelligence," Spectrum, Vol. 20, No. 11, November 1983, pp. 77 - 78.

Feigenbaum, E. A., and P. McCorduck, "Japanese Lead The Field In AI Technology," Microprocessors At Work, Vol. 5, No. 8, June 1984, pp. 1 - 4.

Firdman, H. E., "MIS Can Stimulate Change In Corporate View Of AI," Computerworld, Vol. 20, No. 34, August 25, 1986, pp. 61 - 66.

Gabriel, R. P., "Lisp Expert Systems Are More Useful," Electronics, August 7, 1986, p. 65.

Gaines, B. R., and M. L. G. Shaw, "From Fuzzy Logic To Expert Systems," Information Sciences, Vol. 36, No. 1-2, August 1985, pp. 5 - 16.

Guenthner, F., H. Lehmann, and W. Schonfeld, "A Theory For The Representation Of Knowledge," IBM J. Res. And Dev., Vol. 30, No. 1, January 1986, pp. 39 - 56.

Guterl, F., "Next Generation Impacts," Spectrum, Vol. 20, No. 11, November 1983, pp. 79 - 81.

Haley, P. and C. Williams, "Expert System Development Requires Knowledge Engineering," Computer Design, Vol. 25, No. 4, February 15, 1986, pp. 83 - 86.

Harris, L. R., "When Is An Expert System Right?" Information center, Vol. 2, No. 7, July 1986, pp. 96 - 97, 119.

------, "Integrated Expert Systems," Information Center, Vol. 2, No. 9, September 1986, pp. 23 - 24.

Hayes-Roth, F., "Codifying Human Knowledge For Machine Reading," Spectrum, Vol. 20, No. 11, November 1984, pp. 79 - 81.

------, "Knowledge-Based Expert Systems," Computer, October 1984, pp. 263 - 273.

------, "The Knowledge-Based Expert System: A Tutorial," Computer, Vol. 17, No. 9, September 1984, pp. 11 - 28.

------, "Rule-Based Systems," Communications Of The ACM, Vol. 29, No. 9, September 1985, pp. 921 - 932.

Hayes-Roth, Frederick, Donald A. Waterman, and Douglas B. Lenat, Building Expert Systems, Addison-Wesley, Reading, Massachusetts, 1983, Vol. 1.

Hinden, H. J., "Fifth Generation Computing: Dedicated Software Is The Key," Computer Design, Vol. 23, No. 10, September 1984, pp. 150 - 164.

Jacobson, A. D., "Lisp Is Not Needed For Expert Systems," Electronics, August 7, 1986, p. 64.

Kehoe, C. A., "Interface And Expert Systems For Online Retrieval," Online Review, Vol. 9, No. 6, December 1985, pp. 498 - 502.

Kellog, C., "From Data Management To Knowledge Management," Computer, January 1986, pp. 75 - 83.

Landry, J. B., "Examining Expert Systems," Computerworld, Vol. 20, No. 27A, July 9, 1986, pp. 69 - 71.

Liebowitz, J., "Useful Approach For Evaluating Expert Systems," Expert Systems, Vol. 3, No. 2, April 1986, pp. 86 - 96.

Mackintosh International, Expert Systems 1985-1990, A Report prepared by Mackintosh International Limited, 1985, Vol. I - IV.

MacNicol, G., "Artificial Intelligence And The Fifth Generation," Digital Design, Vol. 15, No. 4, April 1985, pp. 34 - 40.

Mantelman, L., "AI Carves Inroads: Network Design, Testing, And Management," Data Communications, July 1966, pp. 106 - 123.

Manuel, T., "What's Holding Back Expert Systems?" Electronics, August 7, 1986, pp. 59-63.

Michaelson, R. H., D. Michie, and A. Boulanger, "The Technology Of Expert Systems," Byte, Vol. 10, No. 4, April 1985, pp. 303 - 312.

Miller, F. D., J. R. Rowland, and E. M. Siegfried, "ACE: An Expert System For Preventative Maintenance Operations," Record, January 1986, pp. 20 - 25.

Mittal, S., C. L. Dym, and M. Morjaria, "PRIDE: An Expert System For The Design Of Paper Handling Systems," Computer, Vol. 19, No. 7, July 1986, pp. 102 - 114.

Moore, R., "AI Must Cater To Nonexperts," Computer Design, Vol. 25, No. 4, February 15, 1986, p. 68.

Newquist, H. P., "DP Struggles Against Artificial Intelligence Invasion," Computerworld, Vol. 20, No. 27, July 7, 1986, p. 17.

Nofel, P. J., "There's Nothing Artificial About A.I.," Modern Office Technology, Vol. 31, No. 2, February 1986, pp. 40 - 44.

Patton, C., "Making Logic Programs Execute In Parellel Is Today's AI Challenge," Electronic Design, Vol. 32, No. 25, December 13, 1986, pp. 73 - 74.

Philips, R., "Can Fifth-Generation Software Replace Fallible Programmers?' Computerworld, Vol. 18, No. 29, July 16, 1986, pp. 27 - 30.

Rifkin, G., "Toward The Fifth Generation," Computerworld, Vol. 19, No. 18, May 6, 1985, pp. 16 - 23.

Rokey, M., "The Dataflow Architecture: A Suitable Base For The Implementation Of Expert Systems," Computer Architecture News, Vol. 13, No. 4, September 1985, pp. 8 - 14.

Shipley, C. "Execs Probing AI Use Seen Poaching On MIS Turf," PC Week, Vol. 3, No. 22, June 3, 1986, p. 141.

Simons, G. L., "Fifth-Generation Computers: Features And Research," Software World, Vol. 14, No. 4, 1983, pp. 5 - 9.

Sriram, D., and M. D. Rychener, "Knowledge-Based Engineering Systems - Research In Progress," IEEE Software, Vol. 3, No. 2, March 1986, pp. 48 - 60.

Stolfo, S. J., "Knowledge Engineering : Theory and Practice," pp. 97 - 104. Proceedings of Trends and Applications 1983. Automating Intelligent Behavior. Applications and Frontiers, May 25 - 26, 1983, Gaithersburg, Maryland. IEEE : New York, 1983.

Suydam, W. E. Jr., "AI Becomes The Soul Of The New Machines," Computer Design, Vol. 25, No. 41, February 15, 1986, pp. 55 - 70.

Thompson, B. A., and W. A. Thompson, "Inside An Expert System," Byte, Vol. 10, No. 4, April 1985, pp. 315 - 330.

Tou, J. T., "Design Of Expert Systems For Integrated Production Automation," Journal Of Manufacturing Systems, Vol. 4, No. 2, 1985, pp. 147 - 156.

Tsuji, S., and E. H. Shortliffe, "Graphical Access To Medical Expert Systems: I. Design Of A Knowledge Engineer's Interface," Methods Of Information In Medicine, Vol. 25, No. 2, April 1986, pp. 62 - 70.

Turner, M., "Real Time Experts," Systems International, Vol. 14, No. 1, January 1986, pp. 55 - 57.

Walton, P., "Prolog To The Fifth Generation," Informatics, Vol. 5, No. 12, December 1984, pp. 53 - 55.

"What's Happening With Expert Systems," EDP Analyzer, Vol. 23, No. 12, December 1985, pp. 1 - 11.

Williamson, M., "Expert System Shells - Design Tools Help MIS Answer Management's Call," Computerworld, Vol. 20, No. 28, July 14, 1986, pp. 51 - 60.

Zadeh, L. A., "Making Computers Think Like People," IEEE Spectrum, August 1984, pp. 26 - 32.

XI. PERSONAL DATA

Carol Ann Koperna was born October 5, 1957 to Mr. and Mrs. Joseph L. Koperna of Coaldale, Pennsylvania. After graduation from Panther Valley High School in May of 1975, she was accepted by Pennsylvania State University in September of 1975. She graduated in May of 1979 with a Bachelor of Science degree in Computer Science.

Upon graduation she accepted a position with A T & T Technologies, Inc. as an Information Systems Designer and relocated to Allentown, Pennsylvania. In 1981 she was promoted to an Information Systems Staff Member. On September 15, 1984 she married Daniel T. Liscinsky.

She remains currently with A T & T Technologies, Inc. where she now directs a Divisional Software Development Group for the Components and Electronic Systems Division.