# DEVELOPMENT AND IMPLEMENTATION OF AN EXPERT SYSTEM TO DIAGNOSE MOTOR VIBRATION PROBLEMS

by

**James E. Corley**

**Senior Engineering Consultant**

**and**

**Gary D. Darby**

**EDP Capacity Planner**

**ARAMCO**

**Dhahran, Saudi Arabia**

*James E. Corley is presently a Senior Engineering Consultant for the Arabian American Oil Company in Dhahran, Saudi Arabia. He has corporate responsibility for technical matters relating to dynamic analysis and rotating equipment. He joined ARAMCO in 1973, to organize the company's vibration analysis group.*

*Prior to his present employment, he was with Union Carbide Corporation's Nuclear Division in Oak Ridge, Tennessee. There he was responsible for compressor seal development for the gaseous diffusion complex. Mr. Corley holds B.S. and M.S. degrees in Mechanical Engineering from Mississippi State University and has done postgraduate work at the University of Tennessee.*

*G. D. Darby is a computer professional with 25 years of experience on IBM mainframe equipment as programmer, systems programmer, and analyst. He has a B.S. degree in Mathematics from Michigan State University and an M.S. in Engineering Management from the University of Missouri-Rolla. He has been employed as an EDP Capacity Planner at ARAMCO since 1981.*

## ABSTRACT

An expert system, TurboExpert, has been developed to diagnose vibration problems dealing with rotating equipment. Starting with a small inexpensive expert system shell, the program was expanded to include many characteristics required for practical use including the ability to handle uncertainty in rules or facts, to deduce multiple solutions and to modify previously entered facts as a session progresses. The program is designed for use by the inexperienced machinery engineer. It interfaces with the user via menu driven answers and graphic displays of allowable responses to computer generated questions. After completing a query session, the program lists the probable machine faults in descending order of certainty and gives advice about possible corrective actions.

The system begins with an initial goal of diagnosing the cause of a problem. It operates by using deductive inference to derive new facts from known facts and rules contained in a knowledge base. The engineer is queried for information about vibration characteristics when that information is required.

A prototype knowledge base dealing with motors is in day-to-day use in a motor repair shop. The expert system's success rate is currently over 90 percent. A knowledge base dealing with pump problems is under development.

## INTRODUCTION

A great deal of excitement about artificial intelligence (AI) has been generated in the press recently. Millions of dollars are being spent annually on AI research. Robotics, vision systems, and truly intelligent machines hold great promise for the future. But what about today? The problems of emulating human intelligence with a computer are large and far from solution. However, for the engineer facing real world diagnostic problems, one branch of AI, expert systems, can provide a great deal of valuable assistance.

The authors' experience in developing the software and knowledge bases for an expert system to diagnose mechanical problems in rotating equipment is conveyed herein. The system, which is called TurboExpert, was low in cost, runs on common IBM PC hardware, and is providing valuable advice to non-expert technicians on a daily basis at ARAMCO.

TurboExpert is a sophisticated delivery mechanism for implementing diagnostic decision trees. The software technology involved is somewhat complex but achievable. The true power of the system is in the knowledge base. The ability to capture in a few hundred rules and with a few man-weeks of effort what an expert has learned by hard experience over many years, provides tremendous leverage for this technology.

Four major sections are discussed herein: The first provides an introduction to the role of expert systems and introduces terminology and concepts. Next comes the examination of some of the characteristics of real world systems which are commonly missing from the "toy" expert systems available on the market today and which have been implemented in TurboExpert. The current project is described: how the software was developed, problems and advantages of the "do-it-yourself" approach, implementation details, and the current system status at ARAMCO. Finally, guidelines are provided for evaluating the application of expert systems in any company and summaries of current and future directions are presented.

## WHAT IS AN EXPERT SYSTEM?

### *Definition*

Expert systems are computer systems which use knowledge and inference procedures to solve problems which normally require human expertise. The knowledge usually takes the form of heuristic rules and observed facts. Inference is the logical process which combines rules and facts to produce new facts. The intention is to emulate on a computer the process common to all problem solvers, whether they are detectives, mechanics, computer programmers, . . . or engineers.

Mentioning only "hard science" practitioners does not imply that managers, lawyers, and secretaries are not also problem solvers. Today's technology cannot compete with humans in the "soft" problem arenas dealing with social interactions. Common sense is a most difficult area of knowledge to emulate on a computer. One of the surprising results of AI research to date is that fields which traditionally were considered to reflect high intelligence, logic and numerical manipulations, are easy to implement. The "simple" problems (e.g., "go to the store and buy a quart of milk") require vast amounts of knowledge to solve. Fortunately for the current study, engineers deal with hard data and narrow problem definitions which can be emulated quite well.

### *Advantages and Disadvantages*

It is important to realize that today's expert systems are not generalized problem solvers. They can be easily fooled and misled. Protecting the novice user from inadvertently leading the system to give bad advice is one of the challenges facing those implementing expert systems.

Expert systems should emphasize the good qualities of human expert advisers and minimize the undesirable ones. They should also capitalize on what computers do best. Computers, like elephants, have excellent memories and, once given a set of conditions leading to a particular diagnosis, will have no difficulty remembering the case next year. They are very methodical; they don't panic easily; they don't have bad days or arguments with their wives; they don't mind being consulted at 3:00 a.m.; they don't take planned vacations; and they don't retire or repatriate to their country of origin, taking their expertise with them. These provide strong reasons for investigating expert systems in areas that have significant costs associated with expert performance levels.

Before proceeding to a more detailed examination of the program, some basic concepts and terminology will be introduced. These will ease understanding of the system being described, and provide valuable background when the reader proceeds with investigations in his own area.

### *Types of Expert Systems*

There are two common approaches to expert system implementation, both using a "deductive" form of inference. Forward chaining systems move through a rule base deducing all possible facts. Given the rule "IF a THEN b" and the fact "a," it will conclude "b" and add it to a list of known facts. These systems are useful in real time control applications, where all implications of each new piece of data must be investigated. Forward chaining systems are commonly called "production systems," because in each cycle they produce an additional fact. Rules in production systems are called "productions." AI researchers favor production systems because they provide a reasonable model of how humans think.

The second basic method of inferencing, and the one used in this program and most expert systems, is called "goal-directed" or "backward chaining." Backward chaining systems are easier to implement because they are concerned only with facts related to a single specific goal. They start with a goal to be proven and work backwards to resolve it. Backward chaining expert systems direct user sessions by asking very specific questions, such as "Does the vibration amplitude peak during coastdown?" Facts not related to a current specific goal are not normally accepted. (An exception procedure in the authors' system implements "non-monotonic" logic, a technique forcing the system to revise previously entered facts and redirect it's processing.) If a narrow focus and lack of generality is accepted, the goal-directed approach can provide systems which operate efficiently, are easy to write, easy to understand, and provide excellent results in their area of expertise.

### *Inside an Expert System*

The key components and interfaces of a typical backward chaining expert system are shown in Figure 1. It is instructive to "play the game" of emulating a computer emulating a human expert. To begin, imagine a desk with three piles of paper and a telephone on it. The piles of paper represent "memory" contents and the telephone represents a user contact. There is one pile for rules, one for facts, and one for goals. Initially only paper in the rules pile has any writing on it (If-Then rules). A typical real-world rule is presented in Figure 2, although simpler rules will be used in this example.

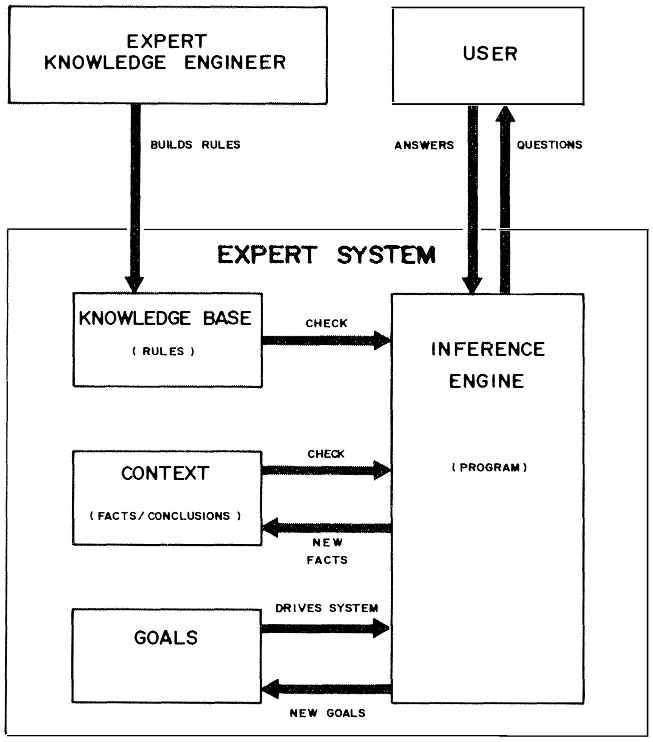

*Figure 1. Schematic Drawing of an Expert System.*

The session begins with a call from a user with a problem. The goal "find problem cause" is recorded on the top sheet of the goals pile. The goal pile is called a stack, which in the computer language simply means that new goals always go onto and come off of the top (a LIFO queue). The pile of known facts is called the current "context."

Facts, rule conditions, and rule conclusions in the system are attribute/value pairs. An attribute/value pair may be thought of as a simple equation, say $X = 47$, where X is the attribute and 47 is the value. The attribute and value in this simple system are connected by a predicate expressing the relationship between the attribute and the value (e.g., "is equal to", "is greater than", etc.). Goals are attributes for which values are to be determined. As goals are resolved, they are moved to the facts pile. The objective is to find a value for the goal "problem cause" and make an entry in the facts pile "problem cause is xxxxx." The task is

SAMPLE RULE

```
 IF        * BEARING TYPE IS SLEEVE
      AND  * VIB SPECTRUM IS COMPLEX
      AND    VIB = 2x SPEED
      AND    VIB = SPEED
      AND    VIB DROP IS RAPID
      AND    HIGH AXIAL VIB IS YES
      AND    BOTH BEARINGS IS NO
 THEN      PROBLEM IS COCKED BEARING        ( 70 )
```

*Figure 2. A Typical Rule Used in a Knowledge Base.*

finished when the goal pile becomes empty. Note that this expert system deals primarily in the world of words rather than numbers. Thus, a typical intermediate fact will be "vibration frequency is equal to motor running speed," and a typical final solution will be "problem is cocked bearing."

Begin the expert system "game" by checking through the fact list to see if the value of "problem" is already known. Probably not. Next, look through the rules pile and find a rule with "problem" in its conclusion part. If there are none, the attribute cannot be resolved except by asking the user. Assume that a rule is found, "IF amplitude is high THEN problem is resonance." Thus, if it could be proven that "amplitude is high," the user could logically deduce a value for the "problem." The next step, then, is to see if the value for amplitude is known. This is accomplished by putting "find amplitude" on top of the goal stack and checking, first the context for a value for "amplitude," then the rule stack for a rule which assigns a value to "amplitude" in its conclusion part. This may lead to additional goals on the goal stack, as the operator chains backwards through the rule base. For example, in this case, a rule "If measured amplitude > 5 mils then amplitude is high" would lead to a new goal, "find measured amplitude."

Whenever a goal cannot be resolved using known facts and rules, the user will be consulted to provide the data. Eventually either the original goal is resolved and the answer can be returned or, if there were not enough rules to handle this problem, the only way to find the problem cause is ask the friendly user (in which case he may not be so friendly).

That's all there is to this simple system. Most low cost expert system shells on the market operate in a similar manner. For a more complete discussion of this methodology as applied in the current system, see William and Beverly Thompson's excellent article [1].

## REAL WORLD CHARACTERISTICS

If the previously described system provided a complete solution, any number of existing low cost systems would have served to meet existing needs. In fact, emulating real world expertise requires a number of features not handled very well or at all by the simple system just described. These desirable features are summarized in Table 1.

For example, in querying the user, it is desirable to minimize bad input due to typing errors. This can be accomplished by providing a list of all valid answers in menu form whenever a question is asked. Such a menu-driven interface has been implemented in this program.

The simple "game" system did not address uncertainty of the user inputs or of the rules. The handling of uncertainty in expert

*Table 1. Desirable Characteristics of a Practical Expert System.*

DESIRABLE EXPERT SYSTEM CHARACTERISTICS

- ○ GOOD USER INTERFACE (MENU DRIVEN)
- ○ CERTAINTY VALUES FOR RULES
- ○ ABILITY TO HANDLE "UNKNOWN" AND UNCERTAINTIES
- ○ ABILITY TO RE-ENTER AND SAVE FACTS
- ○ MULTIPLE CONCLUSIONS RANKED BY PROBABILITY
- ○ ABILITY TO EXPLORE SYSTEM REASONING
- ○ GRAPHICS
- ○ USER WRITTEN FUNCTIONS
- ○ INEXPENSIVE

systems has become a major sub-field for study and will be discussed in more detail. Heuristic rules typically allow operators to "deduce" causes given symptoms. It is rare that a particular set of symptoms points absolutely to the answer. Therefore, it is important to allow the expert to specify rules in the form of, "If I see symptoms x and y, then in my experience there is an n percent chance that the problem is z."

From the user's perspective, an appropriate answer to a question from the system may be "It is likely that the value is y" or "I don't know the answer to that question." Human experts do not usually give up, hang up, or force the user to take an absolute guess under these conditions; computerized versions shouldn't either.

Human advisors also accept revised input if the customer changes his mind or finds additional information later; they may even suggest that the user obtain additional data and call back. When calling back, the user should not have to start over describing the problem. Thus, the ability to re-enter data during a session and to save and restore the facts base for use at a later date is an important expert system characteristic.

Allowing for uncertainties in conclusions implies that multiple causes may be suggested for a particular set of symptoms. The simple system stopped when the first value for "problem" was found. The full expert system incorporates logic which derives all possible solutions from known facts and prints a ranked list of all those with certainties above a specified level.

In addition to diagnosing the problem, a human expert advises the customer about solving it. A simple explanation facility has been included in the program to provide this information.

There are a number of other desirable features relating to ease of use and ease of building/maintaining knowledge bases which will be discussed later.

## THE PROJECT

The authors work in separate divisions of ARAMCO and had independently arrived at the conviction that expert system technologies hold a great deal of promise. As expatriate employees, they are charged with transferring technology to Saudi nationals as rapidly as possible. Expatriates are hired as an intermediate solution to meeting the corporate requirement for expertise. One of the authors has a background in software development and the other represents diagnostic expertise in rotating equipment. This was a happy combination of talents which enabled a workable prototype system to be developed with approximately four man-months of effort.

The driving force for the development of the expert system was to capture the diagnostic skills of the skilled expatriate professional in order to improve the performance of inexperienced personnel. A vibration monitoring program had been in place in the company for several years [2]. This monitoring program consists of separate vibration groups operating in different plants which monitor rotating machinery and analyze the problems which are found. The vibration analyst is generally a fairly experienced expatriate engineer or machinist. However, turnover and attrition is expected to create an experience gap in this area. Therefore, there is a need to transfer knowledge and diagnostic skills through the use of an expert system. The application of an expert system to the area of machinery diagnostics is a natural fit. The logical If-Then rules used by the machinery analyst in diagnosing vibration problems are very similar to those of the medical diagnostics systems use in the early development work in expert systems such as MYCIN.

## *Program Goals*

The primary objective of the program was to create an expert system in the area of machinery diagnostics which would produce credible solutions to field problems with an acceptable accuracy rate. It was felt that for the system to be credible and usable, it should be successful in arriving at correct diagnoses at least 80 percent of the time. Thus, the 80 percent success rate was made a goal of the program.

The secondary objective of the program was that the system should simulate or mimic the response of an expert. This meant that the output of the program should be phrased in a similar manner to what one would expect from the expert. For example, an expert diagnostician will seldom be able to give an unqualified answer to a problem. In most cases, there are several possible causes to a problem with similar symptoms, and he will give the most probable cause, followed by a ranking of less likely conditions. This type of response gives the proponent a better feel as to how to attack the problem and more flexibility in considering his options. He can either address the most likely cause or he may start with the one which is easiest to check.

## *The Inference Engine*

There was no large budget for this project, and systems which would have required hardware beyond the available IBM PCs and PC-XTs were not feasible candidates. The softare base finally chosen was MicroExpert, a low cost expert system shell (inference engine) marketed by McGraw-Hill. This system is written in Turbo Pascal and was designed to run on IBM PC-sized hardware. Since MicroExpert is distributed with source code, it was possible to incorporate additional desirable features into the program. Permission was obtained from the software publisher to distribute a copy of the modified system within the company, for each copy of the original product purchased.

TurboExpert contains more than twice as many lines of source code as the original MicroExpert (4900 lines compared to 2300 lines originally) and makes extensive use of software overlays to keep the code within the Turbo Pascal processing limit (64Kb, or 65,536 bytes). The total compiled code size is about 90Kb with the largest overlay, the inference engine proper, occupying about 52,000 characters of memory. Like MicroExpert, TurboExpert uses linked lists to maintain rules, facts, and goals in addition to several other kinds of data (user prompts, translations for output, etc). Linked lists take advantage of all memory available up the 640Kb limit of the IBM PC.

Disk accessing is used only for initial loading of the knowledge base, loading code overlays as required, loading graphics pictures for some user responses, and saving facts at the end of a session. The system operates nearly as rapidly from floppy disks as from hard or fixed disks. A "knowledge base compiler," which eliminates the requirement to parse rules as they are read, was implemented. This change reduced the time required to load a knowledge base to 1/3 of the original time. A 100 rule "object" knowledge base requires about 20 seconds to load from a hard disk, compared to one minute for the source version. For current knowledge bases of approximately 100 rules, the time spent resolving goals between inquiries to the user is typically less than one second. It is estimated that the program developed will handle knowledge bases with 300 to 400 rules before memory and response constraints become a problem.

## *The Knowledge Base*

The knowledge base is the section of the program with the If-Then rules, which are used to solve the problem. MicroExpert provided an excellent base for expansion. The rule syntax is straightforward and readable. "Prompt" clauses may be associated with attributes in order to produce more intelligible queries to the user. "Translate" clauses may also be included to produce more English-like outputs. User defined functions to perform arbitrary tests may be implemented in condition clauses, and user defined procedures to assign arbitrary values to attributes may be defined in rule conclusions. TurboExpert uses only the originally supplied "COMPARE" function to test numeric values in conditions and a "MATH" procedure to perform computations on conclusion results. Additional functions and procedures may be added in the future as the need arises.

The syntax of the MicroExpert knowledge base has been expanded in a number of ways. "Initial Goals" may be included; data always required will be obtained at the start of a session rather than as the backward chaining process requires it. "Translates" may be specified for attribute/value pairs rather than for attributes only. This allows explanations of results to be produced. "Picture file" names may be specified as attribute values and will produce graphical response choices to the user. Many run time options may have default values specified on a new "Options" statement in the knowledge base. Finally, a rule confidence level may be attached to any rule. The topic of certainty deserves further discussion.

## *Condition Certainty*

To allow the system to handle real world situations, the concept of certainty was introduced. When the user is prompted by a question from the system, he not only can select an answer, but can also assign a likelihood value to the answer. The value system that was selected for the program is based on "certainly true" having a value of 100 and "certainly false" having a value of 0. In this system, "unknown" has a certainty value of 50. Thus, if a user is not completely sure of his answer, he can give it a certainty value of perhaps 80, an intuitive judgment of the odds that his answer is true. A certainty value is thus applied to each condition in the rule being considered. Although the program uses numerical values for certainty, the user is prompted with a menu of English terms which have numerical values. For example, "likely" is assigned a value of 75.

## *Rule Certainty*

To obtain the overall certainty value level for a rule, a fixed confidence level is assigned to the rule by the knowledge engineer/expert. These values follow the same scheme as certainty values, i.e., 100 represents absolute certainty or true. The expert may give a rule a confidence level of perhaps 60 (possible), when a set of conditions leads weakly to the conclusion.

The confidence level for a conclusion is then modified by the certainty value of the conditions as described previously. The individual certainties for the rule condition values are averaged and an algorithm in the program combines this average with the rule confidence level to arrive at the overall certainty value for the conclusion. For example, if the confidence level for a rule is 80, and the averaged certainty value for conditions is 75, then the overall certainty value for the rule is decreased to 65. In laymen's terms, the conclusion changes from "likely" to "possible." If the overall certainty level for the rule is reduced below 60, it is assumed that there is insufficient knowledge to reach any conclusion and the system discards it.

By this method, the ability to answer "unknown" is automatically incorporated into the system. If, in a string of conditions, the answer is not known to one of the questions, it is assigned a certainty value of 50. This value is averaged with the other certainties and, if the overall certainty for the rule is reduced below 60, the rule is removed from consideration. However, there may be situations where the "unknown" does not reduce the certainty to below 60. Then a conclusion can be reached with a decreased probability of being true. This procedure mimics the reasoning process of an expert. Unknown knowledge may not cause him to fail in reaching a conclusion, particularly when it is not a key piece of information, but it will certainly decrease the probability that it is the right answer.

### *Screening Conditions*

During the development of the system and the knowledge base, it was realized that several conditions used in the If-Then rules did not contain information relating to the conclusion sought by the rule, but rather the condition acted as a "screening condition" to prevent the inference engine from asking the user inappropriate questions and to prevent consideration of rules which might not otherwise apply. For example, one probably should not ask questions concerning orbit shapes if there were not displacement probes on the machine. Likewise, the system should not consider anti-friction bearing problems if the machine has sleeve bearings. These screening questions were used extensively in the knowledge base, but had the disadvantage of over-weighting the algorithm for determining rule certainties. One can postulate a situation of a rule which might have nine screening conditions and only one condition which really applied to the conclusion of the rule. If the nine screening conditions were known to be true, the conclusion would be strongly supported regardless of the key condition certainty. It was concluded that certainties for these screening conditions should not be used in calculating the combined certainty of the condition part of a rule. In the TurboExpert knowledge base, these screening conditions are designated by an asterisk and are used to "screen out" rules from consideration, but are not otherwise used in calculating certainties. An example of this is shown in Figure 2.

### *Conclusion Certainties*

The previously described system for handling certainties applies to single rules. In the real world, an expert may arrive at the same conclusion concerning a problem from the use of several different rules. There might be several different rules, for example, concerning how to diagnose a resonance in a shaft. If more than one of these rules support the diagnosis, then the user could be more certain of the conclusion than if only one rule supported it. This concept of supporting evidence increasing the certainty of the conclusion has been incorporated into the expert system. During a session, the system may arrive at the same conclusion via several rules. At the end of the session, the certainties from these are combined in such a way as to increase the certainty of the final diagnosis. If the system has found more than one possible solution to the problem, they are ranked in decreasing order of probability. The process of determining certainty is shown in Figure 3.

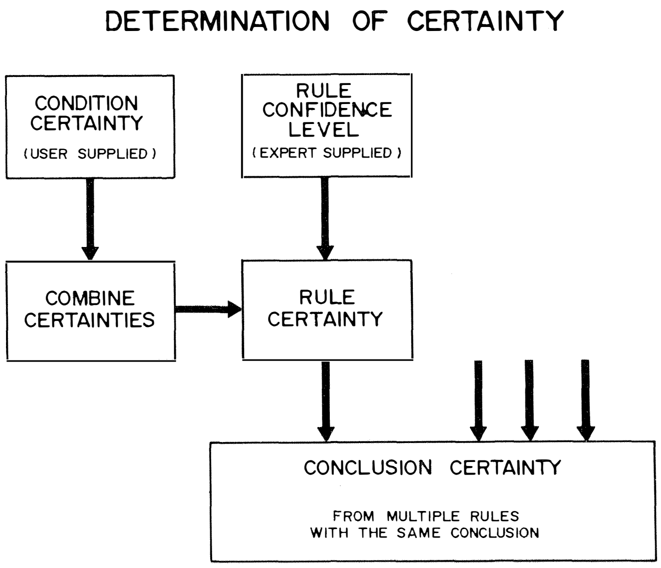
DETERMINATION OF CERTAINTY
CONDITION CERTAINTY (USER SUPPLIED)
RULE CONFIDENCE LEVEL (EXPERT SUPPLIED)
COMBINE CERTAINTIES
RULE CERTAINTY
CONCLUSION CERTAINTY
FROM MULTIPLE RULES WITH THE SAME CONCLUSION

*Figure 3. Schematic Drawing Indicating How Certainty Is Determined.*

### *The Motor Knowledge Base*

The specific area chosen for the first application of TurboExpert was motor diagnostics. This area was picked for several reasons. First, the company depends, to a great extent, on electrical motor drives as the prime mover for much of its equipment. There is a population of approximately 20,000 motors, with sizes ranging up to 30,000 horsepower. This large population, together with the complex electrical/mechanical problems common to motors, results in motor vibration problems being one of the major areas of concern. The second reason for selecting motors for the expert system was the close proximity to the motor repair facilities in the central maintenance shops. The testing of all repaired motors in the shop frequently resulted in calls to the vibration group for assistance. The motor repair facility, therefore, had a strong need to have diagnostic capability, but could not justify a dedicated vibration analyst. The close proximity of the shops meant that there would be the feedback needed in the development of the system. The third reason that the motor shop was selected was the simplicity of shop testing as opposed to field operations. On the test bed, the motor is run uncoupled and unloaded. This greatly simplified the initial operation of the expert system when it was being debugged. The shop environment reduced confusing interactions with coupled machinery and ensured that a problem motor would be inspected immediately.

The initial set of If-Then rules for the knowledge base was based on a diagnostic help chart for motors which was produced by Campbell [3]. This guide furnished an initial set of about 20 rules which related vibration frequencies to common motor problems. This first trial knowledge base only considered horizontal motors with sleeve bearings. It was felt to be important to keep the prototype system small and simple for the first evaluations. Additional rules were added which either did mathematical calculations, such as finding frequency ratios, or set up screening conditions, such as asking questions of the user concerning test conditions. With the addition of these rules, the first prototype knowledge base had about 40 rules.

One of the factors which had to be considered in constructing the knowledge base was that the end users, the shop floor personnel, were not trained in vibration terminology, and the vibration test instrumentation was not as sophisticated as might be found in a troubleshooting or diagnostic group. The spectrum analyzer in use, for example, was a tunable filter meter and not a real time spectrum analyzer (RTSA). This meant that frequency information might not be as accurate as might be desired. Thus, the rules in the knowledge base must make some assumptions such as: if a frequency was plus or minus two percent of the running speed, it was assumed to be once per revolution. Although this might lead to a wrong conclusion some of the time, the knowledge base was built on the premise that it is primarily an aid to the user, and that it might be incorrect occasionally. It attempted, therefore, to use data which might not be exact, but still be correct an acceptable amount of the time.

Another example of this type of reasoning concerned distinguishing between an oil whirl problem and a half frequency problem caused by a nonlinear stiffness. In the oil whirl case, the problem vibration frequency may be close to, but is never exactly, one-half of the running speed. If the frequency information supplied by the user is not accurate enough, as in the shop environment, a half frequency problem might be diagnosed as an oil whirl problem. These problems can be distinguished without a spectrum analyzer if the shaft reference mark on the orbit did not rotate. The knowledge base rules, therefore, ask the user questions concerning orbits and phase reference marks. If this information is unknown, or if an orbit does not exist, then the conclusions reached by the system do not have as high a certainty.

This same reasoning had to be used to determine if an apparent once per revolution vibration was caused by mechanical or electrical forces. To do this, when a RTSA with a zoom feature was not available, rules were required which questioned the user about things such as beat frequencies and rapid drop-off in vibration when the power was cut.

### *Shop Testing*

When the first system was installed in the shop environment, it was put under the care of a skilled motor engineer to use on all motor problems. This was done so that necessary feedback to develop the system could be obtained. To expand the knowledge base and develop the inference engine into a practical system, it was absolutely essential to determine when the system did not find the correct answer to a problem and to find bugs in the programming. This initial feedback from a friendly user in a benign environment is critical to the development of a successful program and cannot be overemphasized.

The initial tests of the limited system in the shop were very encouraging. The major misdiagnoses in the first week of use were blamed by the user more on unfamiliarity with the program than on shortcomings in the knowledge base. These problems were quickly overcome and the use of the system generated an unexpected benefit. Because the system always asked the user the same questions concerning test data and conditions, the shop became much more careful about the testing and performed more consistent tests on the motors. When the user knew he would be queried about the coastdown characteristics of a motor, for example, he started always observing the coastdown. Likewise, when he knew he would be asked about beat frequencies, he started recording them.

This phase of the testing also uncovered bugs in the program. On a problem which the program missed, one concerning a hot spot on the rotor, examination of the pertinent rule and the data available indicated that the program should have reached the correct answer. It was found that a bug in the program was treating an answer of "unknown" as an answer of "no." The correct rule (and conclusion) should have been triggered, albeit with a lower certainty value, instead of failing. This example points out the importance of an expert system having the ability to handle conditions which are classified as 'unknown." In the real world, an expert cannot quit a problem just because he does not know everything possible about the problem. He must of necessity work with partial and incomplete information. The expert system must also continue to execute, if it is to be successful.

After the initial evaluation, the knowledge base was expanded to include rules relating to a broader range of machines and conditions. These rules covered conditions such as running coupled (for later use of the program in the field), motors with anti-friction bearings, and vertical motors. The derivation of rules relating vibration frequencies to anti-friction bearing malfunctions presented somewhat of a problem and is an example of the heuristic reasoning applied to the development of the knowledge base. In general, the engineer or machinist using the program to diagnose a motor problem will not know what bearing is in the machine or the number of balls in the bearing. In addition, a bearing failure which has progressed to multiple flaws does not have the nice, neat frequencies predicted by the design equations. Thus, to include this common malfunction in the knowledge base, a study was made of all the common sizes of bearings used in small National Electrical Manufacturers Association (NEMA) frame motors. The characteristic frequencies of these bearings were calculated and were all found to fall in fairly well defined frequency bands, regardless of bearing size. For example, the ball pass frequency of the inner race fell in the range of 7.15 to 9.8 times the running speed. For practical purposes, it was not necessary, therefore, to know the exact bearing configuration to obtain reasonable and usable results. Experience indicates that it is seldom necessary to know that a spall is on the inner race of a bearing (except to impress a skeptical plant manager). Generally, it is sufficient to know that the bearing is bad and must be replaced. If the expert system can tell the user that there is a probable cause to suspect a spall in the inner race without requiring him to carry around a bearing catalog or open the machine, so much the better.

During the shop evaluation, it became evident that in addition to general rules concerning motor vibration problems, there were also situations which related to specific machines. These were generally design problems with certain models of motors or, more rarely, the design concept pertaining to a manufacturer's entire motor line. Thus, experience had taught that motors from manufacturer XXX, which were above a certain horsepower, had a problem with flexible endshields. It was very easy to include this type of specific information into the knowledge base. Not only would the expert system remember the characteristics and symptoms for such a case, it also could prompt the user to apply a particular work scope which was needed to fix that problem.

Another unexpected benefit was uncovered in shop testing of the system. This was the "mystique" of the computer as seen by the shop personnel. The computer was in some ways more believable than the expert. When a generic problem was found with one of the "problem motors," the machinists were reluctant to offset the rotor in the stator as was required to magnetically load an otherwise lightly loaded bearing. When this repair technique was programmed as a response of the expert system, the fix was accepted without question.

### *Summary of Shop Results*

After the first few teething problems with the system as mentioned previously, the program was moved to the shop floor as a routine tool for the shop personnel to use. The motor engineer was able to transfer the system to the shop mechanics for use on a day-to-day basis. After approximately six months of use, the program was found to produce accurate diagnoses over 90 percent of the time. Although this success rate is higher than the goal of 80 percent, it must be remembered that this evaluation was in a simpler shop test environment. The true test of the system will come in the everyday use in the field.

The user friendly features of the program, such as the extensive use of menus, enabled the noncomputer user on the motor test bed to make rapid progress in applying the expert system to his problems.

The motor knowledge base presently consists of approximately 100 rules and has been given to four vibration groups in the field for their use and evaluation. At present, there has not been sufficient data to estimate the success of the system, although the indications are encouraging. It is expected that with feedback from the field, additions to the knowledge base will make the system smarter and the goal of 80 percent will be achieved.

## RESULTS AND CONCLUSIONS

### *Domain Selection*

Considerable thought has gone into consideration of how to select future domains. The characteristics of the current application which have contributed to its success have been cataloged and are presented in Table 2. This list matches to a large extent one previously published by Prereau [4]. The factors fall into two major categories.

*Table 2. Prerequisites for Successful Expert System.*

**REQUIREMENTS FOR A SUCCESSFUL EXPERT SYSTEM**

- ○ THE DOMAIN IS CHARACTERIZED BY THE USE OF EXPERT KNOWLEDGE.
- ○ THE TASK REQUIRES SYMBOLIC REASONING AND HEURISTICS.
- ○ THERE IS A WILLING EXPERT AVAILABLE.
- ○ EXPERTISE IS NOT AVAILABLE ON A CONTINUING BASIS.
- ○ THE SYSTEM IS EXPECTED TO HAVE A SIGNIFICANT PAYBACK.
- ○ THE FIELD OF KNOWLEDGE REQUIRED IS NARROW.
- ○ THE TASK IS CLEARLY DEFINED.
- ○ THE TASK IS NOT POLITICALLY SENSITIVE.
- ○ THERE IS A FRIENDLY USER.
- ○ THERE IS MANAGEMENT SUPPORT.

First, the task must be appropriate; it must be simple enough and have significant costs (and therefore potential costs savings) associated with it. It should involve expertise based primarily on heuristics and symbolic reasoning. The task should not be on the critical path of some other project. Neither the system, the knowledge it contains, nor the results should be politically sensitive or highly controversial.

Second, the personnel—there must exist an expert willing, able, and available to contribute his knowledge. There should be user area personnel who want the system and are supportive of its development. Finally, management must have recognized a problem and be supportive of the expert system's approach to its solution.

### *Future Directions*

Development work on a pump knowledge base has started and currently has over 110 rules. Work on this knowledge base is expected to take about two months and the system will then be given to the field units for evaluation. The long term objective is to produce expert systems to cover all types of rotating machinery, including compressors, turbines, gears, and seals. It is expected to take approximately one year to accomplish this task.

Current work has convinced the authors that 100 rules is approaching the size limit for developing knowledge bases without specialized tools. Beyond this size, the number of attribute names and rule interactions greatly increases the chance of hidden inconsistencies and knowledge base debugging becomes quite time consuming. A rule editor or checker which can catch unreferenced attributes, inconsistent value assignments, rules which cannot be triggered, etc., becomes increasingly necessary for larger bases.

The explanation facility in the current system is too simple. It merely provides extra text to be printed with a particular attribute-value pair. While this gives the appearance of intelligent responses to the casual user, it would be much more desirable to generate explanations in the same manner that the problem was determined, i.e., from rules. To do this, the system must be capable of interrogating both facts and certainty values. With this facility, key unknown or low certainty facts supportive of a tentative conclusion could be presented to the user as recommendations for further tests.

Access to the historical monitoring database for a machine might allow rules to be written which consider specific machine idiosyncrasies. Direct input of readings taken by digital instrumentation could provide more complete and accurate data than is obtainable from the novice user. Both of these capabilities are on the current "futures" list.

## SUMMARY

Expert systems which provide valuable assistance in areas of hardware diagnostics are achievable today. They can run on personal computer sized equipment and do not require expensive, specialized hardware or software. With consideration of the characteristics of real world problems and the supportive application area personnel, successful systems are readily achievable.

An expert system has been developed to diagnose motor vibration problems in an industrial environment of a motor test floor. This system has met the design goals of the program of being easy to use by the noncomputer user and the novice vibration technician, and it has shown the ability to solve problems at a very succcessful rate of over 90 percent.

## ACKNOWLEDGEMENTS

The authors wish to thank Mr. Peter Edwards of the Mechanical Services Shop Department for his assistance in testing of the system. Thanks are also due Mr. William R. Campbell for his motor troubleshooting guide used in the knowledge base.

## REFERENCES

1. Thompson, W., and Thompson, B., "Inside an Expert System," Byte Magazine (April 1985).
2. Corley, J. E., "A Vibration Monitoring Program Using Microcomputers," Vibration Institute Proceedings (June 1984).
3. Campbell, W. R., "Diagnosing Alternating Current Electric Motor Problems," Vibration Institute Proceedings, pp. 65-79 (May 1985).
4. Prereau, D. S.,, "Selection of an Appropriate Domain," AI Magazine, *6*(2) (Summer 1985).