NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

SPEEDS: AN APPROACH TO SUPPORT PROGRAMMING ENVIRONMENTS USING EXPERT DATABASE SYSTEMS

by

Ronald A. Boxall

September, 1991

Thesis Advisor: Magdi N. Kamel

Approved for public release; distribution is unlimited

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (*If applicable*) 37	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (*City, State, and ZIP Code*) Monterey, CA 93943-5000	7b ADDRESS (*City, State, and ZIP Code*) Monterey, CA 93943-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION 8b OFFICE SYMBOL (*If applicable*)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (*City, State, and ZIP Code*)	10 SOURCE OF FUNDING NUMBERS Program Element No / Project No / Task No / Work Unit Accession Number

11 TITLE (*Include Security Classification*)
SPEEDS: AN APPROACH TO SUPPORT PROGRAMMING ENVIRONMENTS USING EXPERT DATABASE SYSTEMS

12 PERSONAL AUTHOR(S) Boxall, Ronald A.

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (*year, month, day*)	15 PAGE COUNT
Master's Thesis	From To	September 1991	80

16 SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 COSATI CODES			18 SUBJECT TERMS (*continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUBGROUP	Expert Systems, Expert Database Systems

19 ABSTRACT (*continue on reverse if necessary and identify by block number*)

Programming decisions, such as scheduling, planning and corrdinating are made in every type of organization. In situations where these decisions are made by an expert who uses information stored in large databases, it could be advantageous for the organization to employ an expert system coupled with a database to assist in the decision process.

This thesis proposes an approach for building expert database systems to support programming environments. To test this appraoch, a prototype expert database system is developed for a typical programming environment at a classical music radio station that employs experts to select music. The process of acquiring and representing the expert knowledge and the development, testing, and implementation of the prototype are discussed in the context of this case study. The lessons learned in the development of this expert database system are also presented.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION	
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS REPORT ☐ DTIC USERS	Unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL Magdi N. Kamel	22b TELEPHONE (*Include Area code*) (408) 646-2494	22c OFFICE SYMBOL AS/KA

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

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

Approved for public release; distribution is unlimited.

SPEEDS: An Approach to Support Programming Environments
Using Expert Database Systems
by

Ronald A. Boxall
Lieutenant, United States Navy
B.S., The Pennsylvania State University, 1984

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
September, 1991

ABSTRACT

Programming decisions, such as scheduling, planning and coordinating, are made in every type of organization. In situations where these decisions are made by an expert who uses information contained in large databases, it could be advantageous for the organization to employ an expert system coupled with a database to assist in the decision process.

This thesis proposes an approach for building expert database systems to support programming environments. To test this approach, a prototype expert database system is developed for a typical programming environment at a classical music radio station that employs experts to select music. The process of acquiring and representing the expert knowledge and the development, testing and implementation of the prototype are discussed in the context of this case study. The lessons learned in the development of this expert database system are also presented.

TABLE OF CONTENTS

I. INTRODUCTION 1

A. BACKGROUND 1

B. OBJECTIVES 3

C. RESEARCH QUESTIONS 4

D. SCOPE 4

E. METHODOLOGY 4

F. ORGANIZATION OF STUDY 5

II. EXPERT DATABASE SYSTEMS DEVELOPMENT 7

A. DATABASES, EXPERT SYSTEMS, AND EXPERT DATABASE SYSTEMS 7

B. EXPERT SYSTEM DEVELOPMENT METHODOLOGY 10

C. INTRODUCTION TO CASE STUDY 12

III. KNOWLEDGE ACQUISITION 15

A. EXPERT SELECTION 15

B. DOMAIN FAMILIARITY 16

C. KNOWLEDGE ELICITATION TECHNIQUES 18

1. Observation 18

2. Interview 19

3. Role-playing by the KE 20

4. Database Analysis 21

D. KNOWLEDGE ACQUISITION PROCESS - CASE STUDY . . 21

IV. KNOWLEDGE REPRESENTATION 27

A. SELECTION OF A KNOWLEDGE REPRESENTATION SCHEME 27

B. STRUCTURING KNOWLEDGE 28

1. Defining Programming Constraints as Goal Variables 28

2. Using Goal Variable Groups to Formulate Knowledge as Statements 29

3. Conversion of Knowledge from Statements to a Usable Coding Scheme 30

4. The Expert as the Knowledge Engineer 30

C. KNOWLEDGE REPRESENTATION PROCESS - CASE STUDY . 31

V. PROTOTYPE DESIGN, TESTING AND EVALUATION 41

A. PROTOTYPING CONSIDERATIONS 41

B. COUPLING OF EXPERT SYSTEM AND DATABASE 41

C. EXPERT FEEDBACK AND REVIEW PROCESS 42

D. TESTING AND EVALUATION 43

E. DEPLOYMENT AND MAINTENANCE 43

F. PROTOTYPING PROCESS - CASE STUDY 44

VI. CONCLUSIONS AND RECOMMENDTIONS 48

A. SUMMARY 48

B. LESSONS LEARNED 50

1. Finding the Right Expert 50

2. Modification of Existing Database to Support Expert System 50
3. Mustering Continued Support from Management 51
4. "That's the Way it Should Be" Issue 51
5. Expert System Shell Selection 52
6. Other Considerations 52
C. FUTURE ENHANCEMENTS 53

APPENDIX A 55

APPENDIX B 57

APPENDIX C 58

APPENDIX D 69

LIST OF REFERENCES 72

INITIAL DISTRIBUTION LIST 73

I. INTRODUCTION

A. BACKGROUND

Programming decisions are made at all levels of management on a daily basis and are used in every facet of civilian and military life. Programming decisions refer to the technique used by managers to solve problems by optimally allocating scarce resources, such as capital, labor, or time (Cook and Russell, 1989, pp. 32-33). Scheduling, planning, and organizing projects within fiscal constraints are examples of activities that employ these types of decisions. These programming decisions very often are made by managers, or recognized experts in their particular departments, who use their expertise, as well as corporate data, to arrive at their decisions. In many situations, experts need access to large amounts of data in order to make their decisions.

To illustrate an application that uses an expert to assist in programming decisions, one can look at a typical programming problem. An airline company employs an expert who is responsible for scheduling its airplanes. He has a database that contains the information for each route that includes the cities, distance, fuel consumption, demand, fares, and other pertinent information. The expert looks at the data in the database and determines how to maximize his

company's profits through scheduling these aircraft in the most cost-effective manner. The large volume of data that the expert uses gives the expert more possible combinations and options, thereby complicating his decision. Although his solutions are based on many dynamic variables, the expert manages to make good decisions based on his acquired ability and years of expertise.

Although it is usually favorable for an organization to have a resident expert solve its programming decisions, it can be harmful to an organization when that expert has to leave and takes the accrued corporate knowledge with him. The slow, tedious process of training a replacement, if one can be found, usually does not immediately produce an expert of equal quality and may be very costly to the organization.

In the above example, the question arises as to what would happen if the expert decides to leave the company, or leaves for another reason, such as sickness or health. This small airline company can not afford to hire an unqualified replacement, nor is it able to find anyone who has the time to sit down with the expert to adequately extract his knowledge.

In programming environments, such as the one described above, where an expert uses a large amount of data to arrive at a decision, it could be advantageous for an organization to employ an expert system coupled with a database to assist in the decision-making process. With the constant pressure on management to save money and to do more with less, the time

for taking advantage of expert systems is now. Expert database systems that will support programming decisions can be a valuable asset to organizations that make these common types of decisions.

The applicability of these expert database systems to Department of Defense organizations is particularly noteworthy. The military is constantly plagued by the rapid rotation rates of experts who often spend years becoming very proficient at a particular job, only to carry on their expertise to an unrelated job. The expert usually makes an attempt to pass on any acquired knowledge to a newcomer and then moves on to an entirely different position only to take his valuable expertise with him. To harness his expertise of programming knowledge is to improve the entire turnover process and keep knowledge within the organization, instead of losing it with the passing expert.

B. OBJECTIVES

The primary objective of this research is to propose an approach for developing expert database systems to support programming environments. To illustrate the application of the methodology, a secondary objective is to develop a prototype expert database system that assists classical music stations in their everyday task of making music selections.

C. RESEARCH QUESTIONS

The primary research question addressed by this thesis is: "Can organizations that use experts to make programming decisions use expert database systems to improve cost-effectiveness, save time and/or improve the quality of solutions?" The secondary research question is "How could an organization use a commercially available expert system shell, (e.g., VP-Expert), and an existing database to develop an expert database system that supports such decisions?"

D. SCOPE

This thesis develops a methodology for the analysis and design of expert database systems to assist in programming-type decisions. This methodology may be useful to any organization, civilian or military, which makes programming decisions on a regular basis using data contained in corporate databases.

The analysis and design process is presented through the development of a prototype expert database for a classical music station.

E. METHODOLOGY

The methodology used in this research consisted of several steps. First, an extensive literature review was conducted for background information on expert systems, expert database systems and related disciplines, to be used in the development

of our proposed method. Second, an approach was developed to assist users in building expert database systems to support programming decisions. Next, a programming environment was used as a testbed of the proposed approach. The programming environment chosen was a classical music radio station that relied on the expertise of experienced classical music experts to select music to play. Extensive time was spent with the expert reviewing his decision-making process and converting his knowledge to a usable expert database system using the proposed approach. A prototype was developed, tested and eventually implemented at the radio station.

F. ORGANIZATION OF STUDY

The thesis is organized as follows. Chapter II discusses the relevance of this research to programming environments and proposes a method to design an expert database system to support those environments. This chapter also introduces the environment used, as a case study, to demonstrate the applicability of the proposed method. Chapter III presents the Knowledge Acquisition process and applies it to the case study. Chapter IV introduces the process of representing expert programming knowledge and uses the case study to demonstrate the process. Chapter V discusses the prototyping, testing and evaluation and how it is accomplished in the case study. Chapter VI summarizes the main points of the thesis

the thesis and discusses lessons learned as well as future enhancements to the prototype.

II. EXPERT DATABASE SYSTEMS DEVELOPMENT

This chapter provides an overview for the remainder of the thesis. It reviews the concepts of database systems, expert systems, and expert database systems as well as introduces the case study used to demonstrate the proposed approach.

A. DATABASES, EXPERT SYSTEMS, AND EXPERT DATABASE SYSTEMS

Databases are an integral part of virtually every major organization. They are used by banks, government agencies, corporations, advertisers and almost all other types of organizations and provide information on every aspect of our lives. Data in these databases is used for several purposes. In some cases, the data is used for simple storage, manipulation, and retrieval of data, for example, billing data, mailing lists, etc. In other cases, data is used by experts as the basis for making expert decisions. Through this process, database management systems have allowed decision-makers to apply their expertise, or knowledge, to these sometimes sizable amounts of data to arrive at a decision. Every time the database is updated, the expert is faced with a new set of data to analyze. The new or revised data may or may not affect his decision, but it must be considered by the expert.

Expert systems are computer systems that attempt to replicate what experts normally do (Mockler, 1989, p. 100). Specifically, they are systems that model the decision-making environment, and make decisions based on various inputs, such as sensors, consultation, or database information. These systems can provide acceptable solutions in the absence of an expert and become a corporate repository for knowledge that can be modified by an organization as more knowledge is acquired.

Expert database systems are expert systems that use database information as inputs, in order to simulate the knowledge of an expert. Databases are used to store, manipulate and retrieve large amounts of data, while expert systems store the corporate knowledge on how the data is to be used. Usually the expert system and the database are loosely coupled in order to ensure that the database manages the data and the expert system manages the knowledge independently.

Expert database systems could assist in performing the function of the expert, or pass the original expert's knowledge on, at least partially, to provide consistency to an organization. These systems can also be maintained more readily due to their ability to have dynamic parameters modified quickly without the snowball effect associated with traditional programming methods.

Not all programming environments are conducive to an expert database system. There are characteristics that must

exist in a problem domain that signal a particular environment's suitability for an expert database system. The following list represents traits of applications that are best suited for this type of system:

1. Expert decisions are made in a relatively redundant manner (Rolston, 1988, p. 142).
2. Large amounts of data could cause the expert to make a lower-quality decision.
3. A database, or a large amount of data that could be stored in a database, is already being used by the expert in arriving at his decision(s).
4. A suboptimum response is acceptable (Rolston, 1988, p. 142).
5. The expert's knowledge is relatively scarce (Rolston, 1988, p. 142).
6. An expert may not always be available, or be subject to high turnover rates.
7. There is a significant difference between the best and worst performers of the task (Liebowitz, 1988, p.25).
8. The task takes from a few hours to a few weeks to solve (Liebowitz, 1988, p. 25).

The above list demonstrates the applicability of expert database systems to Department of Defense programming environments. There are numerous instances in the military where an expert uses a large amount of data to arrive at a programming-type decision. For example, in the scheduling of major inspections for naval units, various type commanders, Commander Naval Surface Forces Pacific (COMNAVSURFPAC), for example, must keep track of information on a large number of

ships and their associated dates of interest in order to determine when to schedule a certain inspection. These dates include deployment date, already-scheduled inspections, underway dates, etc. Given the relative scarcity of ship-inspection schedulers in the Navy, and the fact that naval officers have a very high rotation rate, an expert database system could definitely assist in this type environment.

B. EXPERT SYSTEM DEVELOPMENT METHODOLOGY

In designing an expert system it is necessary to perform the following steps: Knowledge Acquisition, Knowledge Representation, Prototype Development, Testing and Evaluation, and Deployment (Chorofas, 1987, p. 108). Each of these steps in the expert system development process, depicted in Figure 1, is discussed in detail in Chapters III through V.

In the first step, Knowledge Acquisition, a Knowledge Engineer (KE) attempts to understand the domain of expertise. This can be the most difficult stage of the development process as the Knowledge Engineer's knowledge, rather than the expert's, is actually reflected in the expert system (Rolston, 1988, p. 157). Next, the understood knowledge is represented in a structured format that can be used to organize the decision process. The represented knowledge is also more understandable to the machines that will process the knowledge.

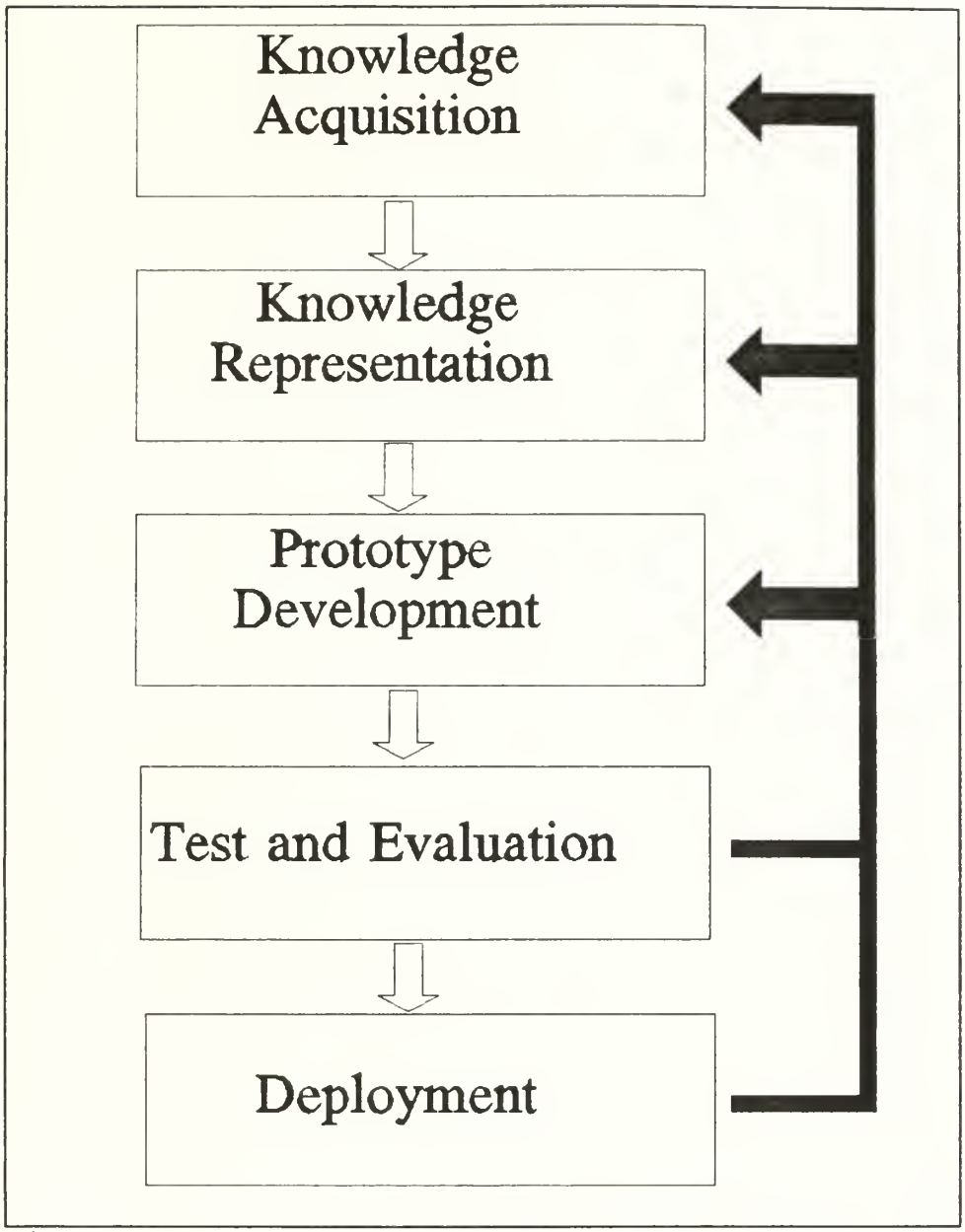

Figure 1. Expert System Development Process

After the expert's knowledge is understood and represented in some scheme, a prototype system is designed to clarify user requirements. A prototype system is a small version of the expert system designed to test assumptions about how to encode

the facts, relationships and inference strategies of the expert. It is the basic building block which is constantly modified during the development process. (Harmon and King, 1985, p. 201)

Once the prototype is acceptable to the KE and the expert, the expert system is tested and evaluated in an effort to ensure that the system fulfills the user's requirements. If the system is considered to be useful, the expert database system is fully implemented. Otherwise, it is iteratively revised until it adequately models the expert's decision process or is abandoned if unable to do so.

Assuming that the system is sufficiently representative of the expert's decision process, it is considered to be useful and is deployed. At this point, it is important to continually review and maintain the system. Just as an expert is always learning, an expert system must be updated to ensure that it still conforms to user requirements, and that the expert knowledge is still valid.

C. INTRODUCTION TO CASE STUDY

To demonstrate our proposed methodology, we apply it to the design of an expert system to make music selections for a classical music radio station. The environment used to apply our methodology is a small radio station with approximately twenty full-time employees: a station manager, a business manager, two engineers, four salespersons, three full-time and

four part-time air personalities, and a small support staff. The senior air personality is also the music director and is responsible for the approval of the station programming schedule.

The music selection process at the station is relatively simple. The air personalities are on the air, usually for six-hour blocks, and selections are made in advance for the entire block. All air personalities select their own music from a large library of approximately 2500 CDs, tapes and LP record albums. All program schedules are subject to approval by the music director and are usually submitted a day or two before the program is to air. Although there are some time periods dedicated to feature programming, approximately 75% of the air time is dedicated to relatively routine selections.

The problems faced by the music station are typical programming decision problems similar to those faced by many organizations. Air personalities with expertise in programming classical music are scarce, particularly in a small geographical area. The result was that cardinal rules of classical music scheduling were violated by less experienced air personalities. Consequently, quality control of scheduling was slipping, and the station was unable to get ahead of its programming schedules. It was also noticed that there were many selections that were not ever being played, or not played often enough. The air personalities were bored

with the tedious process of pre-scheduling their music and sometimes left gaps in schedules.

This situation placed a greater burden on the music director who, in addition to scheduling his own music, was required to spend an increasing amount of time on other less-experienced programmers' schedules. The other air personalities, who enjoyed creating special feature shows, lacked motivation to diversify the everyday programs which took a long time to prepare. The station manager was faced with the problem of trying to maintain a high quality of music scheduling on a relatively strict operating budget.

It was decided that an expert system would be developed to assist the air personalities in making programming decisions by capturing the expertise of the music director and accessing a music library database. The expert system would alleviate the problems associated with having air personalities with varying levels of expertise and would, therefore, require a less experienced staff. Also, the expertise of the popular air personalities could translate into increased listenability ratings, which could ultimately result in increased station revenues.

III. KNOWLEDGE ACQUISITION

The first stage in the development of an expert database system is the Knowledge Acquisition (KA) phase. The KA phase attempts to transfer the knowledge of the expert to the KE, and involves a tedious attainment of domain understanding by the KE. In this phase the KE will ensure that an appropriate expert is selected who can fulfill the needs of the development process, and then endeavor to collect information that will allow him to model the expert's proficiency.

The success or failure of an expert system is based almost entirely on the ability of the knowledge engineer to accurately model the expertise of the expert (Rolston, 1988, p. 120). A KE should be an effective communicator, demonstrate a general competence with the knowledge domain, but should never presume to command the expertise of the expert (Rolandi, 1986, p.47).

A. EXPERT SELECTION

A first necessary step in the KA phase is the expert selection. In some cases, the only available expert will be the one to be used. If the problem-solving technique is similar among a group of experts, one expert could be used for the development and a consensus of experts may be used to

critique the subsequent prototype. If more than one expert exists, the following traits should be used for selecting one for the project:

1. Availability for the length of the project.
2. Willingness to fully support the project.
3. Patient Demeanor.
4. Support of management.
5. Ability to dedicate needed time to KE. (Harmon and King, 1985, p.199)

The above qualities of an expert will support a process of knowledge elicitation that can be tedious, time-consuming and sometimes fruitless.

B. DOMAIN FAMILIARITY

Before initiating an interaction with the expert, the KE must first become basically familiar with the domain so as not to alienate the expert at the outset with questions that demonstrate an obvious lack of knowledge. A mutual respect must be gained between the KE and the expert to ensure a mutual willingness to transmit knowledge. (Rolston, p. 158, 1988)

The most important aspect of this phase is the analysis of the expert's function. In other words, the KE should focus on the data that is required for the expert to arrive at his

decision. The KE should concentrate on understanding the knowledge domain and learning what data is used repeatedly by the expert in the performance of his assigned function. This information will be useful to the KE in interfacing the database with the expert system.

The KE must gain a general familiarity with the domain of expertise of the expert before he can expect to understand the nuances of domain understanding that make the expert a valuable commodity. This is accomplished by the KE becoming part of the background of the expert's environment. The KE should spend as much time as necessary observing the environment of the expert and interviewing people who work with the expert, if they are available. Their perception of the expert and what he or she does could provide a good starting point in the understanding of the expert's knowledge domain, and help the KE understand the working relationship with his co-workers (i. e., what information the expert gains from his co-workers).

It has also been found to be useful to interview other experts, provided they solve the problems in a similar manner, and other personnel who may be responsible for the performance of the job at hand.

Once an understanding of the expert's environment is gained, the KE must become acquainted with the terms, concepts, and "lingo" that the expert deals with in his decision-making process. This information can come from co-

workers, management and others in the field. It is advantageous for the KE to obtain as much information as possible from others who work with the expert not only for the future benefit of knowing what the expert is talking about, but also in order to prevent alienating the expert in the early phases of expert knowledge elicitation.

C. KNOWLEDGE ELICITATION TECHNIQUES

The third step of the KA phase is the elicitation of knowledge based on the observation of the expert in his familiar environment. After a reasonable understanding of the expert's knowledge domain is accomplished, the KA should now attempt the most difficult, time-consuming aspect of knowledge acquisition. The KE must interact with the expert to gain a specific understanding of the manner in which the expert makes his decisions. Some useful methods of eliciting knowledge from the expert are summarized in the following sections.

1. Observation

The observation of an expert performing his function will provide the KE with a starting point for eliciting expertise. The KE should attempt to observe the expert as he is performing his designated tasks and without any intervention by the KE, and preferably without the expert knowing that the KE is even observing him. A list of questions should be generated by the KE that will later be answered by the expert. Questions such as "What were you

doing when..." or "How did you decide which...," or "Why did you ..," will give the KE insight as to the expert's decision-making process. After observing the expert without intervention, the KE should then do a more detailed observation. This will require the KE to monitor and understand every step of the decision process. Whenever a question comes to mind, the KE will interrupt the expert and obtain an answer to his question. Also, any information that the expert writes down, types in, or otherwise records should be fully examined by the KE. Although this is a very laborious task, it is a very important process and one that will begin to teach the KE the expert's decision process (Hoffman, p.19, 1989).

2. Interview

The interview process can be an excellent medium to transfer expert knowledge to the KE. It can, however, be an uncomfortable experience for the expert and provide misleading answers to questions that are asked in an improper manner. After the detailed observation of the expert, the KE should review all of his notes and determine a general question outline. This question outline should include topic questions of a general nature, the answers to which will undoubtedly spawn more questions.

As important as the questions are the manner in which the questions are asked. The questions should be free of bias

from the perspective of the KE. They should not lead the expert to answer in a certain way. Consider the following question to the classical music station expert programmer:

> "The violin is the preferred instrument to play after a piano piece, right?"

The above example could automatically eliminate other choices that the expert would make. The question should be rephrased to let the expert provide the answer himself:

> "What would you say would be the preferred instrument or instruments that you would play after a piano piece?"

This question leaves the expert with room to give a more knowledgeable answer. The KE should make every attempt to limit biased questioning.

3. **Role-playing by the KE**

After the observation and detailed interviewing of the expert, the KE should attempt to simulate the expert in a role-playing exercise. The KE should actually try to perform the job of the expert while the expert corrects him at every opportunity. The KE's initial attempts would normally be inaccurate. However, the more the KE runs through the exercise, he should start to get a good feel for the expertise of the expert. When the KE feels that he has a good general

understanding of the expertise, he should then represent the knowledge in a useful manner.

4. Database Analysis

The advantage of designing an expert database system is that the database itself will provide the KE with the information that is used by the expert in the decision process. The expert and the KE should review the database to identify those object attributes that are used by the expert to arrive at a decision, in a given situation.

D. KNOWLEDGE ACQUISITION PROCESS - CASE STUDY

The primary expert selected for the development of the classical music expert database system was the station music director. He was very supportive of the project, as he thought that this type of system would generally improve the quality of programming and reduce the time he spent reviewing less-experienced programmers' schedules. The music director was supported by management for the same reasons. There was concern, however, over whether the music director, already overloaded, would have time to support the project. Other experts were also used to provide background knowledge and verify the acquired knowledge.

Domain familiarity was difficult for the KE. Although he had a general understanding and appreciation of classical music, it was a slow process of learning its nuances. The KE

frequently tuned in to the station in order to gain an appreciation for the station's mode of operations.

Initially, time was spent with other air personalities to gain a basic knowledge so as not to alienate the music director at the outset. Additionally, other station personnel were interviewed in order to obtain information about the expert's environment. For example, the KE found that some basic knowledge about the way the expert air personalities selected their music was derived from a casual interview with the radio station manager who, although not a recognized expert, was able to provide some good background information. This information was helpful in giving the KE a starting point for expert knowledge elicitation.

In order to evaluate the function of the music director, the KE attempted to find out what data the expert used and from where it was obtained. The KE walked through the music library, reviewed the running station logs of what had already been played, and concentrated on the music database. Although a computerized database was not in place, a manual system was in place that kept the basic information on each selection, and information that was used by the music director was actually written on the disk sleeves. There was also a color scheme that corresponded to the category that the selection fell into (i. e., symphony, concerto, etc.). Scheduling logs were reviewed in an attempt to reveal procedures and trends.

Once the KE was comfortable with the domain of classical music and the function of the music director, the knowledge elicitation phase was conducted. The methods used to elicit expert knowledge were a combination of observation, interview, role-playing and database analysis. The KE actually was in the studio during live broadcasts and noted everything that the expert wrote down, logged or used to make a selection. After a few sessions of quiet observation and asking few questions, the KE began asking such questions as "What did you write down on the back of that CD cover and why?" and "How come this selection is longer than the last one; was that a conscious decision, or not?"

The questions posed to the music director during this stage of interviewing were very diverse. The KE attempted to understand the meaning of items jotted down in the studio, the order of selections, the classification of classical music and the methodology for selecting one selection over another. The KE was surprised to find out that at many times no rules at all were used in making a selection. The expert simply selected a piece the fit into the time slot available. This is one of the reasons for the station wanting to use an expert system: to provide a more consistent selection methodology for times when air personalities would need to make a selection quickly, for example, when music was not pre-scheduled.

Once the interview process was completed, the KE attempted to emulate the expert, that is, make appropriate selections

based on the expertise elicited from the expert. This was met with limited success at first, but, as the expert's selection process became clearer to the KE, selections were made that were acceptable to the expert.

The final stage of the KA process was to analyze the database to identify which attributes of a piece are employed by the expert to make a selection. The radio station did not have a database at the outset of the project. They did, however, have a physical collection of data, the music library, where data was stored. The station air personalities would write on the CD or LP cover information that is used in making subsequent selections, such as date the selection was last played, whether or not it was a popular piece, the mood of the piece, the period, etc. In essence, the album covers were a manual database similar to a Rolodex file or other manual scheme. A conceptual data model was built using an object-oriented technique to aid the KE in the design of the expert system.

The primary objects were determined to be DISK, SELECTION, CONDUCTOR, and COMPOSER. The DISK object was the medium that actually contained the SELECTION. The DISK could be a CD, an LP or a tape. Each disk is uniquely identified by its type (LP, CD or TP) and Disk Number. The COMPOSER and CONDUCTOR objects were very similar in that they represented an individual who either composed a certain selection or conducted a piece. The SELECTION object was the central

object of the database. In addition, the attributes of the SELECTION object were of primary interest to the experts in making selections. Each SELECTION was uniquely identified by the Disk Type, Disk Number, and a Selection Number, which was the order of the selection on the disk. For example, the fourth piece on the compact disk number 243010 would be identified as CD-243010-4. Figure 2 shows the developed object diagram for the classical music radio station application. Appendix A gives a detailed description of all the attributes of the SELECTION object and an explanation of the acceptable values of those attributes.

The final two attributes of the SELECTION object, mood and listenability were not originally attributes of the station's manual database, but were determined to be necessary qualities that the expert felt should be included. These two attributes also differ from the other attributes in that they require an expert's subjective rating. These knowledge attributes are an important characteristic of expert database systems and require an expert evaluation to assign a value to the attributes.

The expert uses other attributes in the SELECTION object to arrive at his decision as to which selection he will play on the air at any given time. The characteristics that the expert determines are pertinent to the selection process are Mood, Instrument, Period, Performing Group Size, Listenability, Disk Number, Minutes, Key and Composer. In

other words, these are the attributes that must meet some selection criteria or are subject to programming restrictions.

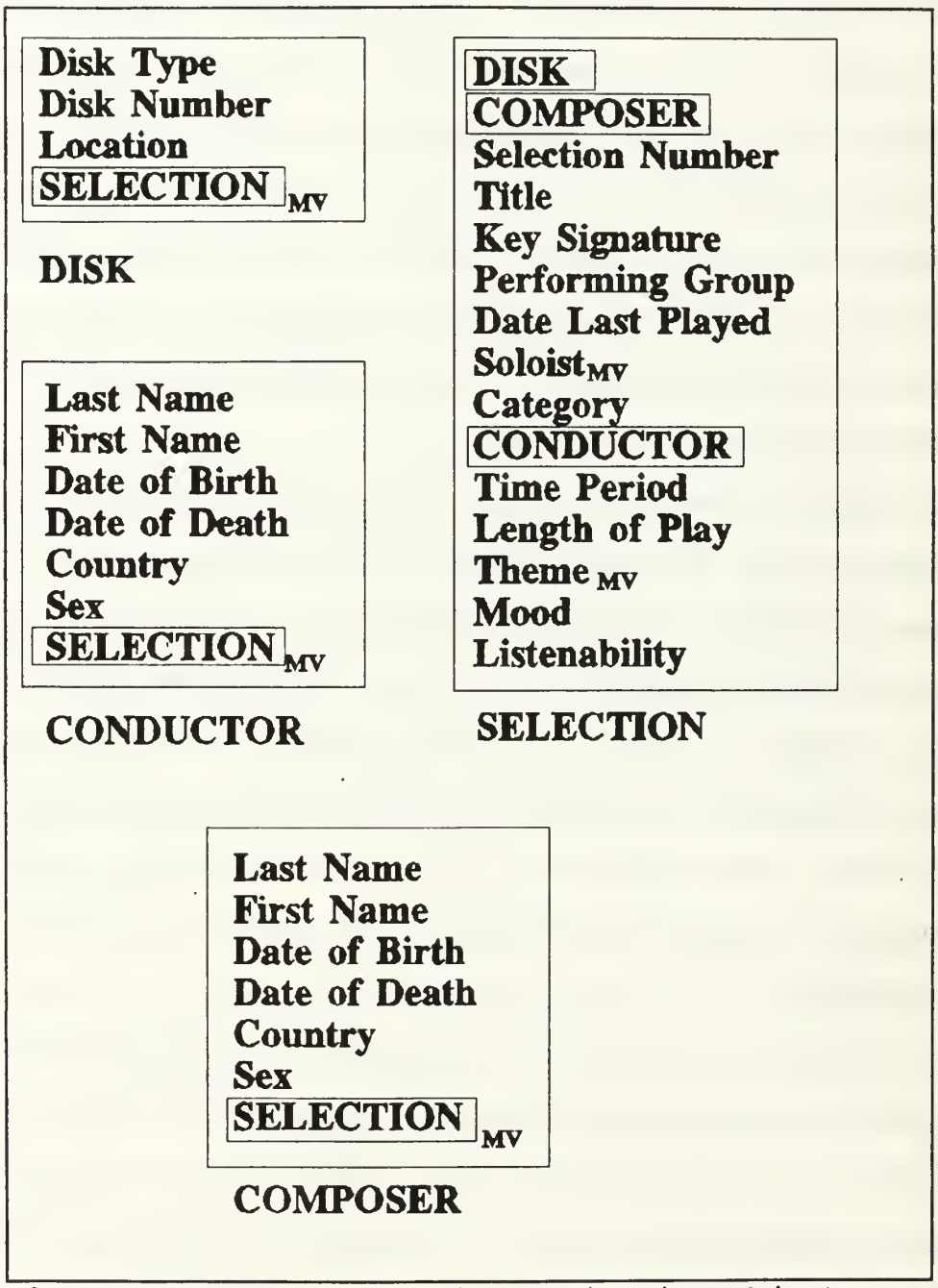

Figure 2. Classical Music Application Objects

IV. KNOWLEDGE REPRESENTATION

The second phase of expert database system development is the depiction of knowledge through some representation scheme. There are many different schemes that can be used. Issues in the representation of knowledge, however, are common to all. These include formulating the knowledge as a statement, and coding it in some scheme (Rolston, 1988, p. 32). This chapter discusses the procedures associated with formulating knowledge from programming environments into a structured scheme.

A. SELECTION OF A KNOWLEDGE REPRESENTATION SCHEME

There are many ways to represent knowledge: semantic networks, rules, frames, and logical expressions are the most common. Each scheme has advantages and disadvantages. (Harmon and King, 1985, pp. 35)

Semantic networks are a collection of objects, or nodes, connected by arcs, or links. The advantage of this scheme is its flexibility. It is easy to show that a certain object, an arm, for example, has a number of two in the average person. Problem arise with this schema when exceptions occur such as when a person has only one arm. (Harmon and King, 1985, pp.35-35)

Rules are commonly used in expert systems due to their ease in representing human thought processes. Rules consist of a premise and a conclusion, or an IF-Clause and a THEN-Clause. The premise must be true in order for the conclusion to be true. The disadvantage to using rules is that it is difficult to model complex knowledge. (Harmon and King, 1985, p. 42)

Frames, which are very similar to semantic networks, are objects that consist of sets of slots which contain properties associated with the frame object. For example, if COAT were a frame, slots could be Condition, Owner, Size, etc. Each of these slots would contain an entry such as worn, John, Size 42, etc. Frames are useful to use in environments where exceptions are rare; they can be inadequate to represent knowledge when exceptions are more common.

Logical expressions use connectives such as AND and NOT to represent relationships. The only values returned by a logical expression are true and false. It is a powerful approach to representing knowledge but is more complex. (Harmon and King, 1988, p. 46)

B. STRUCTURING KNOWLEDGE

1. Defining Programming Constraints as Goal Variables

In programming decisions, constraints are frequently placed on the characteristics of the selection to be made. These constraints include time, weight, and cost, to name a

few. Making a selection, in a programming decision, is to attempt to satisfy all constraints. If this is not feasible, constraints are relaxed until a suboptimal solution is reached.

In designing an expert database system for a programming environment, it is necessary to identify all the characteristics of a particular object instance that fulfill all the requirements of its selection. These characteristics are referred to as goal variables. The goal variables for the classical music station are those attributes that are pertinent to the selection process. Many of these goal variables are included in the music library database. Others are non-database attributes, such as Time of Day. These goal variables, once identified, give the KE a framework for representing the knowledge as statements that select values for some or all goal variables. Therefore, it is mandatory that the KE strive to identify these goal variables and the rules used by the expert to identify their values.

2. Using Goal Variable Groups to Formulate Knowledge as Statements

The knowledge that has been captured by the KE up to this point has been largely unstructured, mainly as brief notes, questions and answers, and unwritten recollection of the KA process. To gain the advantage of machine processing, the KE must begin to structure this captured knowledge into a

collection of statements that will later be converted to a coding scheme.

To start the statement formulation process, the KE must begin converting his notes to statements that fall into specific goal variable groups. For each statement, the KE should ask the question "What goal variable group does this knowledge fall into?" All unstructured knowledge should fall into one of these groups. If it does not, there is a missing goal variable that must be identified.

3. Conversion of Knowledge from Statements to a Usable Coding Scheme

The most common way to represent knowledge, when the expertise is gained entirely from a human expert, is through the use of procedural, or IF-THEN rules. Expert systems that use procedural rules are known as production systems. (Hayati and Parker, 1987, p. 779) In this scheme, a series of IF-THEN rules are created based on the statements developed in the statement formulation phase.

The next step for the KE is to take the statements for each goal variable and structure them in a format dictated by an expert system shell, or programming language.

4. The Expert as the Knowledge Engineer

As the KA phase progresses and the expert can see his knowledge represented as rules, he gradually becomes the KE, formulating his knowledge as rules without intervention by the

KE. This is actually the best possible situation, as the problems associated with interpreting the expert's expertise is gone. (Rolston, 1988, p.167)

C. KNOWLEDGE REPRESENTATION PROCESS - CASE STUDY

In selecting a knowledge representation scheme to be used for the classical music example, it was determined that since the representation scheme most common with human expertise is the rule-based approach, this scheme was used for the classical music application.

The goal variables for the classical music station are those characteristics that were determined in the database analysis to be pertinent to the selection process in addition to non-database goal variables, such as Time of Day. The goal variable groups, as identified are Mood, Instrument, Period, Performing Group Size, Listenability, Composer, Category, Disk Number, Selection Length, Key, and Time of Day. For example, the KE has determined that the Mood is a characteristic of a selection to be chosen. The KE must then decide to write down, as statements, how the expert goes about finding the Mood. In this case the KE first asked the question "How does the expert determine the Mood of the selection he will eventually play? The answer to that question will provide statements which will be grouped under the Mood goal variable group. The following are examples of how knowledge was

acquired from the music director and converted to the following statements and goal variable groups:

Mood Group

- Selections should follow the "dayparting" concept, that is, the music should correspond to activities that the listener would be doing at that time of the day. This means softer music during the dinner and waking hours, and harsher in the later evening hours.

Instrument Group

- If an instrument is featured in one selection, the next selection should not include the same instrument, unless a specific instrument is intended to be highlighted during a planned music set (e. g., The Piano Hour).
- No vocal music should be played except between 8pm and midnight.
- Certain instruments should not follow other instruments.

Period Group

- A selection's time period should be determined by the previous selection's time period, namely that it should be within at least two periods of the previous piece.

Performing Group Size Group

- The "texture" of the selection should not vary too greatly in successive selections.

Listenability Group

- A selection that is well-known should start off the top of the hour, as a "grabber."

Category Group

- No selection should be followed by a selection of the same category

Disk Number Group

- Successive selections should not come from the same disk.

Selection Length Group

- The selection's length shall be determined by the available time in the program block. It must allow for a few seconds between selections.

Key Group

- A selection, such as a symphony or a concerto, should be followed by a selection of a different key if a key is specified for the selection.
- Certain dissonant keys should only be played during the evening hours.

Time of Day Group

- Certain periods of the day are set aside for a particular mood of music.

The above statements, elicited from the expert, are then converted into rules in the format required by the expert system shell. This procedure is illustrated in the following examples:

Mood Group

Statement: "Selections should follow the "dayparting" concept, that is, the music should correspond to activities that the listener would be doing. This means softer music during the dinner and waking hours, and possibly harsher in the later evening hours."

```
RULE 100
IF sked_hour >= 6 AND sked_hour <8
THEN
rmood = Soft

RULE 105
IF sked_hour >= 8 AND sked_hour <12
THEN
rmood = Soft
rmood = Med

RULE 110
IF sked_hour >= 12 AND sked_hour <18
THEN
rmood = Soft
rmood = Med

RULE 115
IF sked_hour >=18 and sked_hour <20
THEN
```

```
rmood = Soft

RULE 120
IF sked_hour >= 20 AND sked_hour <=22
THEN
rmood = *

RULE 125
IF sked_hour >=22 AND sked_hour <=24
THEN
rmood = Soft
rmood = Med

RULE 130
IF sked_hour >= 0 AND sked_hour < 06
THEN
rmood = *
```

In this example, sked_hour is the hour that the selection will be played. The mood goal variable is determined by this group of rules. When the expert system attempts to make a selection, the mood of the chosen selection will be either Soft; Soft and Med; or Soft, Med and Harsh depending on the time of day.

This same procedure is applied to each of the other statements in every group.

Instrument Group

Statement: "A selection should not include the same instrument as the previous one, unless a specific instrument is highlighted during a block (or set period of scheduled music)."

Rule:

RULE 150

IF previnst=Piano AND custom<>yes AND custom_var<>instrument

THEN rinst=Violin

rinst=Flute

rinst=Brass

rinst=Trumpet

rinst=Oboe

rinst=Strings

rinst=Guitar

rinst=Harp

rinst=Percussion

rinst=Organ

rinst=Winds

rinst=French_Horn

rinst=Clarinet

rinst=Cello

Statement: "No vocal music should be played except between 8pm and midnight."

Rule:

RULE 101

IF sked_hour> 8pm AND sked_hour < 12 mid

THEN rinst=vocal

In reviewing the first rule of this group, one can see that the premise for the rule includes multiple conditions. The conclusion for this rule, the instantiation of the instrument goal variable, will only be true if the instrument featured in the previous selection is the piano, and only if this is not a customized program for instrument (or a piano-featured program).

In the second rule, a vocal piece will only be selected between certain hours. Considered together, both rules indicate that the Instrument goal variable can be a vocal piece, during certain hours, or any other instrument, except the previous one, at all other times.

Period Group

Statement : "A selection's time period should be within at least two time periods of the previous piece."

Rules:

RULE 200

IF prevper=Classic

THEN

```
rperiod=Early
rperiod=Baroque
rperiod=Romantic

RULE 210
IF prevper=Baroque
THEN
rperiod=Classic
rperiod=Romantic
rperiod=Early

RULE 220
IF prevper=Romantic
THEN
rperiod=Classic
rperiod=Modern
rperiod=Baroque
rperiod=Contemporary

RULE 230
IF prevper=Modern
THEN
rperiod=Classic
rperiod=Romantic
rperiod=Contemporary
```

```
RULE 240
IF prevper=Contemporary
THEN
rperiod=Modern
rperiod=Romantic

RULE 250
IF prevper=UNKNOWN
THEN
rperiod=*
```

In the above example, Rule 200 states that if the previous selection was from the Romantic period, then Period Goal Variable could only be Classic, Modern, Baroque or Contemporary.

Listenability Group

Statement: "Well-known selections should start off the top of the hour, as a grabber, otherwise the popularity doesn't really matter."

```
Rule:       RULE 500
            IF sked_minute>=0 AND sked_minute<3
            THEN
            rlisten=High
            ELSE
```

rlisten=*

This example provides another capability of rules, the capability to provide an alternative if the premise is false. In this case, only at the top of the hour, if the minute after the hour is zero through three, the Listenability Goal Variable must be High.

The process of developing rules continued for every goal variable group. When all the rules were specified, the next phase, prototype development, began.

It is interesting to note that during the KA phase, once the expert became familiar with seeing his knowledge represented as rules, he began expressing his knowledge as IF-THEN statements. In this case, the expert was gradually becoming the KE.

V. PROTOTYPE DESIGN, TESTING AND EVALUATION

A. PROTOTYPING CONSIDERATIONS

As soon as the KE has formulated the expert's knowledge as rules, even though a thorough knowledge of the expert's task was not acquired, he should develop a prototype. The prototype will further clarify the expert's knowledge to the KE. The prototype can and should be developed as soon as possible in order to save wasted effort of the KE by potentially pursuing inaccurate knowledge statements.

In developing the prototype for an expert database system, the rules that have already been formulated in the knowledge representation phase should be implemented in the prototype.

B. COUPLING OF EXPERT SYSTEM AND DATABASE

In building the prototype, the KE must determine how to couple the expert system with the database. As most expert system shells are not very efficient in their manipulation of data, the KE should ensure that database operations are handled by calls to the database management system (DBMS), for increased efficiency. Ideally, the DBMS should be used to access and manipulate the data (i. e., sort, filter, etc.) prior to being called by the expert system. Likewise, the DBMS should be used for the ordinary update, modification, or deletion of records, as that is the main function of the

database. Figure 4 represents the loosely coupled relationship between the expert system and the database. This loose coupling gives the organization a multi-use database; one that can be used for the ordinary record-keeping, information storage, and other database functions, and one that can be used by an expert system as a basis for decision-making. It must also be noted that the expert system could have the ability to alter the database if this is a requirement of the expert system; however, normal update functions should rest with the DBMS.

C. EXPERT FEEDBACK AND REVIEW PROCESS

The development of the prototype is an important process for the KE to demonstrate his interpretation of the expert's knowledge. Invariably, demonstration of the prototype to the expert will cause the KE to gain a better appreciation for the manner in which the expert goes about making his decisions. The prototype will either accurately reflect the expert knowledge, in which case the KE can tie up the loose ends of the development process, or else the prototype will not be what the expert had in mind, in which case the KE must modify the prototype to concur with the expert's interpretation. A third possibility also exists. The prototype demonstrated that the effort will be unable to be developed in a manner that is sufficient (within time constraints, budget restraints, etc.).

D. TESTING AND EVALUATION

When the prototype is completed with the database and the expert is satisfied that the knowledge base is consistent with his expertise, the system should be tested with actual data, and its conclusions reviewed thoroughly by the expert. Although the expert may, in some situations, arrive at different solutions, the expert system should generate solutions that are deemed acceptable by the expert. Once the testing with the expert is accomplished, the system should be tested on a small test set of potential users. Although the users may not have sufficient expertise to determine whether or not the conclusions are correct, they will be able to provide useful feedback on their perception of the system. Any recommended changes should be reviewed by the KE and the expert, and the necessary changes incorporated.

E. DEPLOYMENT AND MAINTENANCE

After successful implementation of the system, the system should be deployed to all end-users. Modifications from this point on should be centrally managed in order to ensure that any changes to the knowledge base, or changes in the way that the expert makes his decision, are fully documented and distributed. The expert may or may not be available after the deployment of the system, therefore it is imperative that the system be constantly reviewed for accuracy. Database administrators should be informed as to how the expert system

relates to the database and attempt to prevent modification to the database that could affect the operation of the expert system. Documentation should be provided along with a central point of contact for matters concerning the expert database system.

F. PROTOTYPING PROCESS - CASE STUDY

In developing the prototype for the classical music application, it was first necessary for the KE to consider the coupling between the expert system and the database. The basis for database integration with the expert system is via the GET statement. In VP-Expert, the GET statement retrieves the first record from the database that meets all the criteria of the goal variables. Consider the following statement:

```
GET minlength <= minutes AND maxlength >=minutes AND
prevdisk <> disknum AND rmood=mood AND rperiod=period
AND rlisten=listen AND rperfsize=perfsize AND
rinst=insttype AND rlisten=rlisten AND
rcategory=category AND rkey=key,selects,ALL
```

In this statement, a database record that satisfies all goal variable values determined by the expert system will be retrieved. If the expert system determines that a "Soft" selection is to be played, then the database will be searched

for records that have a "Soft" mood. Adding many dynamic combinations of moods, instruments, categories, etc. narrows down the selection to a manageable size. In coupling with the database, it was important to include all goal variables in the GET statement and to make sure that those goal variables are assigned values.

After coupling was considered by the KE, the first prototype was presented to the expert and revealed that the KE did sufficiently represent the knowledge of the expert. The KE did not, however, accurately reflect the way in which the expert actually performs his function. Specifically, the prototype system was designed to select only one musical piece at a time, while the expert wanted a system that would pick all the selections in a certain "block" of time, usually one to six hours long. The prototype had to be modified to allow the selection of multiple pieces for a varying block of time.

Also during the prototype review phase, the expert modified some of the previous rules. For example, a rule specified earlier by the expert revealed that a Romantic piece could be followed by a selection from any other period. Upon examining the selections of the expert system, the expert felt that a Romantic piece should only be allowed to be followed by a Classic, Baroque, or Modern selection. Therefore, the following old rule:

```
RULE 220
IF prevper=Romantic
THEN
rperiod=Classic
rperiod=Modern
rperiod=Baroque
rperiod=Contemporary
```

was modified to:

```
RULE 220
IF prevper=Romantic
THEN
rperiod=Classic
rperiod=Modern
rperiod=Baroque
```

It was also determined, for increased system flexibility, to add new rules that will allow the users to select specific composers, instruments or other variables, during certain time blocks. An example of the VP Expert statements that accomplish this is:

```
RULE 2050
IF custom_var=Instrument THEN
FIND sinst
```

```
        which_inst=selected
    ELSE sinst=*
        which_inst=not_selected;
```

In the above rule, the user has the option to customize the Instrument variable, otherwise, the usual knowledge-based selection for the instrument goal variable is performed.

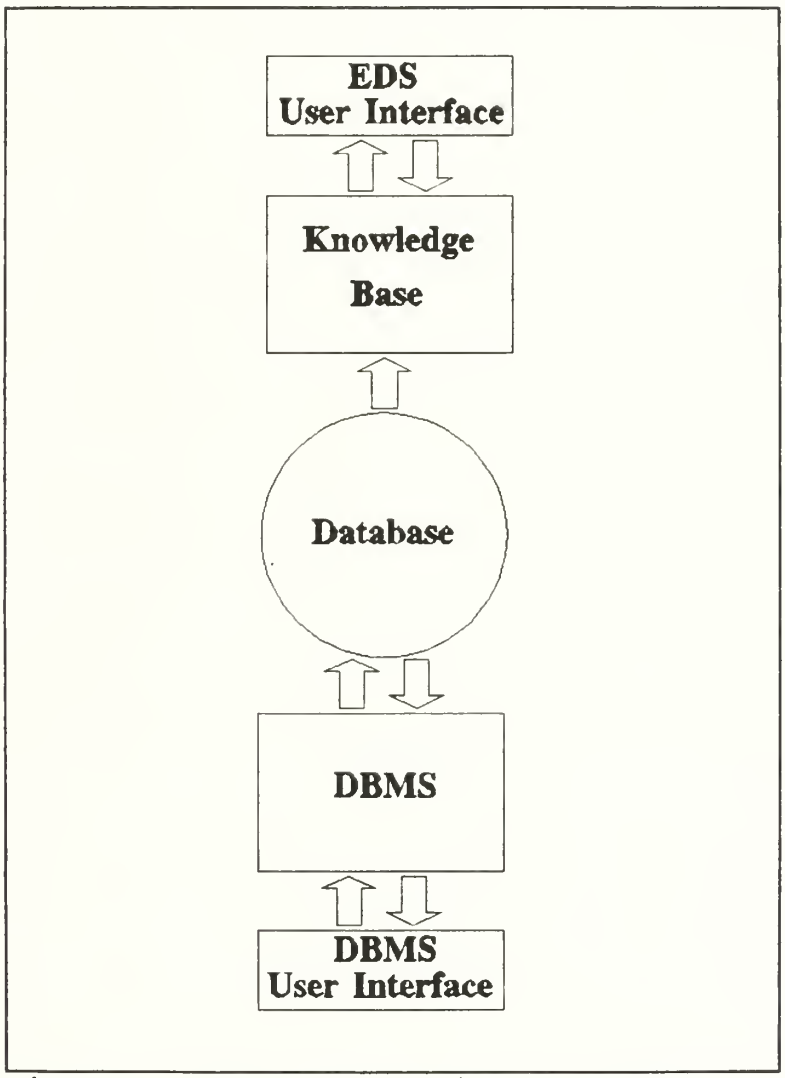

Figure 3. User Perspectives of Expert Database System

VI. CONCLUSIONS AND RECOMMENDTIONS

A. SUMMARY

Experts in all types of organizations make programming decisions based on information contained in databases. Expert database systems which assist experts with their decision making process can save valuable expert time, improve the quality of decisions, and save money. It has been demonstrated that, in the case study, this approach has been useful in developing an expert database system that has the potential of making better decisions faster, saving money, increasing morale and increasing the quality of music at the radio station. These benefits can be enjoyed by organizations that take advantage of these types of systems.

With the constant move toward automation, fewer experts with corporate knowledge will remain. Therefore, organizations should use expert database systems to harness expert knowledge, where possible. The approach presented is ideal for small to medium applications where experts are a valuable asset. The cost can initially appear to be high due to the time that an expert can dedicate to such a project, but in the long run, with continued management support, these systems can result in higher productivity rates for the users of the expert database systems (Rolston, 1988, p.255).

First, an expert and a KE who can sufficiently fulfill the needs of the project must be chosen. Once selected, the expert(s) and the KE should undergo a tedious knowledge transfer phase where the KE will attempt to understand the logic behind the expert's decision process. The knowledge must be represented as English statements prior to being encoded in some scheme. This coding scheme will then be used to develop a prototype to be reviewed and tested by the expert. Once reviewed, the prototype can either be accepted, modified or terminated. If accepted, the prototype is then prepared for deployment and use by the organization. The system must then be maintained and modified as necessary as the knowledge base ages, or as new knowledge is gained.

In the case presented, the expert database system is more cost effective, saves time and improves the quality of programming scheduling for routine selections. The process of programming a six-hour block that would take up to two hours without the system, can now be accomplished in less than two minutes. The expert database system is free from human bias, which will allows it to make selections on the entire database instead of limiting itself to favorites. The experts, who generally dislike the selection process, were very receptive to the system because they felt that they could spend more time programming feature blocks.

B. LESSONS LEARNED

There were many lessons that were learned in the course of developing the prototype. These lessons were learned at various stages of the development process and should be considered by organizations attempting to develop expert database systems.

1. Finding the Right Expert

This may seem like a relatively easy task, however, organizational politics makes this task difficult, especially where more than one expert exists. For example, one expert may be offended and become hostile to the effort by not being selected in the development of the system. Management should assist in the selection of an expert after careful consideration.

2. Modification of Existing Database to Support Expert System

As indicated earlier, many attributes of an object are used solely by experts in arriving at a decision. These attributes are often not included in the existing database. For example, in the developing the prototype of our case study, the existing database did not include some attributes that the expert uses in selecting a piece, such as Mood and Listenability. The KE must be capable of modifying an existing database or designing a new database, if necessary,

to satisfy the needs of the expert system accessing the database.

3. **Mustering Continued Support from Management**

It was found in developing the prototype that management was very motivated at the outset of the project. After the project started and the slow process of acquiring knowledge was being performed, management began to lower the priority of the project, and as a result, time with the expert was somewhat harder to obtain. Frequent updates by the KE to the senior management personnel would be very helpful in maintaining a high level of motivation for the project.

4. **"That's the Way it Should Be" Issue**

Upon review of past decisions, it was noticed that many selected pieces did not follow the logical reasoning that had been explained and demonstrated to the KE. When asked for a reason, the expert explained that in the past, some decisions were made quickly and without much thought, but that the decision process passed on to the KE was "the way it should be." The lesson learned is that the KE should be aware that decisions made in the past may not have taken all rules into account, and experts are sometimes liable for making mistakes. A good example of this situation was the scheduling of two piano pieces that were played consecutively, although the expert explained that no two pieces of the same instrument should be played consecutively. The expert explained that he

was forced to make this particular decision in a short time, and as a result, did not follow the same thought process that he would go through normally. Expert database systems are very helpful in this regard, as the thought process can be modeled in a more relaxed environment, and the knowledge stored in a knowledge base. This knowledge base may provide a more thorough thought process in a time-constrained situation.

5. **Expert System Shell Selection**

VP-Expert had many features that were advantageous to this application, specifically its low cost, use of rules to create knowledge base, and microcomputer development capability. It also had some very difficult shortcomings, such as its inability to directly access Paradox database files, in Paradox format, poor numerical manipulation and limited use of memory. There are many expert system shells available and considerations other than cost should be reviewed prior to choosing one.

6. **Other Considerations**

It was the case in the development of the classical music station prototype that the primary expert was reluctant to let the knowledge engineer discuss the project with other experts at the station. These psychological factors must be

carefully considered and discussed with top management prior to the selection of the expert.

C. FUTURE ENHANCEMENTS

There are many enhancements that could be added to the classical music station expert database system. In the future, the system could be expanded to select an entire week's worth of programming, instead of the block approach of up to 24 hours. There is also room to modify the knowledge base to include specific programming blocks (i. e., new scheduling approaches). In the future, the station may decide to feature a specific instrument, composer or theme during a particular hour every week. The nature of the knowledge base lends itself to this simple type of modification.

The current expert database system could also be enhanced in an effort to minimize the amount of licensing fees paid by the station. Presently, the station pays a flat rate for licensing fees. It is possible to pay much less if an accounting of when licensed selections are played could be provided, and if those licensed selections are only played at certain times of the day. The database could be modified to allow for a field that designates whether or not a selection is licensed. If it is licensed, then that piece should be played at times when the licensing fees are lower, such as nights and weekends. This enhancement would be relatively

simple to add, yet could save the station thousands of dollars per year in royalties paid to licensing companies.

APPENDIX A

DESCRIPTION OF FIELDS IN **SELECTION** OBJECT

Disk Number	Consists of a two-letter media code (LP or CD) followed by a six-digit number corresponding to the disk's assigned reference number. Example: CD-200006
Selection Number	The sequential selection on the disk. Example: 01
Title	The title of the individual selection. Example: Sonata in C Major
Composer	First and Last Name of the primary composer. Example: Ludwig von Beethoven
Key Signature	The primary key in which the selection is played. Example: B Flat
Performing Group	The size of the group that performed the selection. Acceptable values: Orchestral, Ensemble, Solo, Chamber, Other.
Date of Last Play	The last date that the selection was played in the format mm/dd/yy. Example: 01/14/91
Soloist	First and Last Names of up to four soloists. Example: Thomas Weinstein
Instrument	The instrument featured in the selection Example: Piano
Category	The general category of music.

	Acceptable values: Overture, Symphony, Concerto, Ballet, Tone Poem, Suite, Solo, Chamber Piece, Other
Conductor	The first and last name of the conductor of the performing group. Example: Arthur Fiedler
Period	The time period of the selection. Acceptable values: Early, Baroque, Classic, Romantic, Contemporary, Modern
Length of Play	Total playing time of the selection in minutes and seconds. Example: 21:14
Mood	Subjective quality of the mood of the selection based on expert opinion. Acceptable values: Soft, Med, Harsh
Listenability	Subjective quality of the popularity of the selection based on expert opinion. Acceptable values: 1 (popular) to 5 (Obscure)
Theme	Theme of the selection. Example: Patriotic

APPENDIX B

Definition of Goal Variables and Acceptable Values

Database Goal Variables:

Mood	Soft, Med, Harsh
Performing Group	Orchestral, Ensemble, Solo, Chamber, Other
Instrument	Piano, Organ, Strings,Violin, Viola, Cello, Bass, Brass, Trumpet, Winds, Clarinet, Flute, Bassoon, Percussion, Harpsichord
Selection Length	Minutes and Seconds Available.
Category	Overture, Symphony, Concerto, Ballet, Tone Poem, Recital, Suite, Solo, Chamber Piece, Other
Period	Early, Baroque, Classic, Romantic, Modern, Contemporary
Listenability	Rating from 1 (popular) to 5 (obscure).

Non-Database Goal Variable:

Time of Day	The hour and minute that the selection will be played. Input to expert system during consultation.

APPENDIX C

Expert System Program

```
runtime;
ACTIONS

MENU stheme,ALL,selects,Theme1

FIND block_start_hour
FIND block_start_minute
FIND block_length
FIND custom

block_minutes_remaining= (block_length)
block_seconds_remaining= 0
block_minutes_used     = 0
block_seconds_used     = 0

sked_hour=(block_start_hour)
sked_min=(block_start_minute)
sked_sec=0

prevmedia=(media)
prevdisk=(disknum)
prevselnum=(selectnum)
previnst=(insttype)
prevkey=(key)
prevperfsize=(perfsize)
prevcat=(category)
prevper=(period)
prevtitle=(title)

WHILETRUE block_minutes_remaining > 0 THEN

  WHILEKNOWN minutes

    RESET rmood
    RESET rinst
    RESET rperiod
    RESET rperfsize
    RESET rcategory
    RESET rlisten
    FIND rmood
    FIND rperiod
    FIND rinst
```

```
    FIND rperfsize
    FIND rcategory
    FIND rlisten

    minlength=0
    maxlength=(block_minutes_remaining)

    RESET get_clause
    FIND get_clause

!   GET STATEMENT

    RESET message
    FIND message

    prevmedia=(media)
    prevdisk=(disknum)
    prevselnum=(selectnum)
    previnst=(insttype)
    prevkey=(key)
    prevperfsize=(perfsize)
    prevcat=(category)
    prevper=(period)
    prevtitle=(title)

    block_minutes_remaining  =  (block_minutes_remaining  -
    minutes)

    block_seconds_remaining  =  (block_seconds_remaining  -
    seconds)

    block_minutes_used     = (block_minutes_used + minutes)
    block_seconds_used     = (block_seconds_used + seconds)

    sked_sec=(sked_sec+seconds)
    sked_min=(sked_min+minutes)

  END

  RESET message2
  FIND message2

END
;
!==================== Start of Rules ============!

RULE 100
IF sked_hour >= 6 AND sked_hour <8
THEN
rmood = Soft
;
```

```
RULE 105
IF sked_hour >= 8 AND sked_hour <12
THEN
rmood = Soft
rmood = Med
;

RULE 110
IF sked_hour >= 12 AND sked_hour <18
THEN
rmood = Soft
rmood = Med
;

RULE 115
IF sked_hour >=18 and sked_hour <20
THEN
rmood =soft
;

RULE 120
IF sked_hour >= 20 AND sked_hour <=22
THEN
rmood = *
;

RULE 125
IF sked_hour >=22 AND sked_hour <=24
THEN
rmood = Soft
rmood = Med
;

RULE 130
IF sked_hour >= 0 AND sked_hour < 06
THEN
rmood = *
;

RULE 150
IF  previnst=Piano AND custom<>yes AND custom_var<>instrument
THEN
rinst=Violin
rinst=Flute
rinst=Brass
rinst=Trumpet
rinst=Oboe
rinst=Strings
rinst=Guitar
rinst=Harp
```

```
rinst=Percussion
rinst=Organ
rinst=Winds
rinst=French_Horn
rinst=Clarinet
rinst=Cello

RULE 200
IF prevper=Classic
THEN
rperiod=Early
rperiod=Baroque
rperiod=Romantic
;

RULE 210
IF prevper=Baroque
THEN
rperiod=Classic
rperiod=Romantic
rperiod=Early
;

RULE 220
IF prevper=Romantic
THEN
rperiod=Classic
rperiod=Modern
;

RULE 230
IF prevper=Modern
THEN
rperiod=Romantic
rperiod=Contemporary
;

RULE 240
IF prevper=Contemporary
THEN
rperiod=Modern
rperiod=Romantic
;

RULE 250
IF prevper=UNKNOWN
THEN
   rperiod=*
;

RULE 260
```

```
IF sked_hour >= 6 AND
   sked_hour < 9
THEN
  RESET rperiod
  rperiod=Classic
  rperiod=Baroque

BECAUSE "Baroque and Eggs is a special program that
includes only Baroque and Classical Music between 6 and 9 AM"
;

RULE 270
IF sked_hour >= 6 AND sked_hour <9 AND prevper=Classic
THEN
 RESET rperiod
 rperiod=Baroque
;

RULE 275
IF sked_hour >= 6 AND sked_hour <9 AND prevper=Baroque
THEN
 RESET rperiod
 rperiod=Baroque
 rperiod=Classic

BECAUSE "Baroque and Eggs is a special program that includes
only Baroque and Classical Music between 6 and 9 AM."
;

RULE 300
IF previnst=UNKNOWN or rinst=UNKNOWN
THEN
rinst=*
;

RULE 400
IF prevcat=UNKNOWN
THEN
  prevcat=*
;

RULE 500
IF sked_min>=0 AND sked_min<3
THEN
rlisten=High
ELSE
rlisten=*
;

RULE 600
IF prevperfsize=Solo
```

```
THEN
rperfsize=Ensemble
rperfsize=Chamber
rperfsize=Other
;

RULE 610
IF prevperfsize=Ensemble
THEN
rperfsize=Solo
rperfsize=Chamber
rperfsize=Other
;

RULE 620
IF prevperfsize=Chamber
THEN
rperfsize=Orchestral
rperfsize=Other
;

RULE 630
IF prevperfsize=Orchestral
THEN
rperfsize=Chamber
rperfsize=Other
;

RULE 640
IF prevperfsize=UNKNOWN OR prevperfsize=Other
THEN
rperfsize=*
;
RULE 1000
IF custom=No THEN
  get_clause=1

GET minlength <= minutes AND maxlength >=minutes AND prevdisk
<>   disknum   AND   rmood=mood   AND   rperiod=period   AND
rlisten=listen AND rperfsize=perfsize AND rinst=insttype,
selects,ALL
;


RULE 1500
IF custom=Yes THEN
 FIND multiple
 FIND custom_var;
```

```
RULE 2010
IF custom_var=Period THEN
   FIND speriod
        which_period=selected
   ELSE speriod=*
        which_period=not_selected
;


RULE 2020
IF custom_var=Composer THEN
   FIND scomplname
   FIND scompfname
        which_composer=selected
   ELSE scomplname=*
        scompfname=*
        which_composer=not_selected
;

RULE 2030
IF custom_var=Category THEN
   FIND scategory
        which_category=selected
   ELSE scategory=*
        which_category=not_selected
;

RULE 2040
IF custom_var=Theme THEN
   FIND stheme
        which_theme=selected
   ELSE stheme=*
        which_theme=not_selected
;

RULE 2050
IF custom_var=Instrument THEN
   FIND sinst
        which_inst=selected
   ELSE sinst=*
        which_inst=not_selected
;

RULE 2100
IF multiple=No AND custom_var = Period THEN
   get_clause=1
   FIND speriod
```

```
    GET minlength <= minutes AND maxlength >=minutes AND
    prevdisk <> disknum AND rmood=mood AND sperіod=period AND
    rlisten=listen AND rperfsize=perfsize AND rinst=insttype,
    selects,ALL
;

RULE 2200
IF multiple=No AND custom_var = Instrument THEN
   get_clause=1
   FIND sinst

    GET minlength <= minutes AND maxlength >=minutes AND
    prevdisk <> disknum AND rmood=mood AND rperiod=period AND
    rlisten=listen AND rperfsize=perfsize AND sinst=insttype,
    selects,ALL
;

RULE 2300
IF multiple=No AND custom_var = Composer THEN
   get_clause=1
   FIND scomplname
   FIND scompfname
    GET minlength <= minutes AND maxlength >=minutes AND
    prevdisk <> disknum AND scomplname=cmplname AND
    scompfname=cmpfname, selects,ALL
;

RULE 2400
IF multiple=No AND custom_var = Category THEN
   get_clause=1
   FIND scategory

    GET minlength <= minutes AND maxlength >=minutes AND
    prevdisk <> disknum AND rmood=mood AND rperiod=period AND
    rinst=insttype AND rlisten=listen AND scategory=category,
    selects,ALL
;

RULE 2500
IF multiple=No AND custom_var = Theme THEN
   get_clause=1
   FIND stheme

    GET minlength <= minutes AND maxlength >=minutes AND
    prevdisk <> disknum AND rlisten=listen AND stheme=theme1
    OR stheme=theme2, selects,ALL
;

RULE 2900
IF multiple = yes THEN
   get_clause=1
```

```
   FIND which_period
   FIND which_composer
   FIND which_category
   FIND which_inst
   FIND which_theme

    GET minlength <= minutes AND maxlength >=minutes AND
    prevdisk <> disknum AND rmood=mood AND speriod=period AND
    scomplname=cmplname    AND    scompfname=cmpfname    AND
    scategory=category AND stheme=Theme1 AND stheme=Theme2 AND
    sinst=insttype, selects,ALL
;

RULE 3000
IF minutes <> unknown
THEN
   message=displayed

    DISPLAY "At time {2sked_hour}:{2sked_min}:{2sked_sec},
    selection is {media}-{disknum} #{selectnum} by {cmplname}.
    {period},Mood={mood},Listen={listen},size={perfsize},{in
    sttype},cat={category},{3minutes}:{2seconds}"

ELSE
   message=none
;

RULE 3100
IF minutes = UNKNOWN AND block_minutes_remaining>0 OR
block_seconds_remaining >0
THEN
   message2=displayed

    DISPLAY "There is insufficient data fill remaining
    {block_minutes_remaining}   minutes   and
    {block_seconds_remaining} seconds."

ELSE
   message2=none
;

WHENEVER 4000
IF sked_sec>=60
THEN
sked_sec=(sked_sec-60)
sked_min=(sked_min+1)
;

WHENEVER 4100
IF sked_min>=60
THEN
```

```
sked_min=(sked_min-60)
sked_hour=(sked_hour+1)
;

WHENEVER 4200
IF sked_hour>=24
THEN
sked_hour=(sked_hour-24)
;

WHENEVER 4300
IF block_seconds_remaining < 0
THEN
block_seconds_remaining = (block_seconds_remaining + 60)
block_minutes_remaining = (block_minutes_remaining -1)
;


WHENEVER 4400
IF block_seconds_used > 59
THEN
block_seconds_used=(block_seconds_used-60)
block_minutes_used=(block_seconds_used+1)
;

ASK block_start_hour: "What hour will the block start (00-23)
?";
RANGE block_start_hour:0,23;

ASK block_start_minute: "How many minutes after the hour will
the block start?";
RANGE block_start_minute:0,59;

ASK block_length: "How long (in minutes) is the block of time
you want to fill?";

ASK custom:"Would you like to customize this block of music?";
CHOICES custom:Yes,No;

ASK custom_var:"Which of these variables would you like to
modify?";
CHOICES custom_var: Period, Composer, Category, Theme,
Instrument;

ASK multiple:"Would you like to select more than one variable
to customize?";
CHOICES multiple: Yes, No;

ASK speriod:"Select the period to be featured during the
block";
```

```
CHOICES speriod: Early, Baroque, Classic, Romantic, Modern,
Contemporary;

ASK sinst:"Select the instrument(s) you would like to feature
in this block";
CHOICES sinst:  Piano, Strings, Brass, Guitar, Organ, Winds,
Harpsichord, Clarinet, Oboe, Flute, Cello, Bass, Violin;

ASK scomplname:"Enter the Last name of the composer to
feature: ";
ASK scompfname:"Enter the First name of the composer to
feature: ";

ASK scategory:"Select the Category you would like to feature:
";
CHOICES scategory: Symphony, Concerto, Ballet, Tone_Poem,
Vocal, Chamber_piece, Solo, Overture, Recital, Other;

ASK stheme: "Select the Theme you would like to feature: ";

PLURAL:rmood,period,rperiod,custom_var,sinst,rinst,rcategory
,rlisten,rperfsize;
PLURAL:scategory,speriod;
```

APPENDIX D

SAMPLE SESSION OF EXPERT DATABASE SYSTEM

```
Vhat hour will the block start (00-23) ?
01

low many minutes after the hour will the block start?
00

low long (in minutes) is the block of time you want to fill?
60

ould you like to customize this block of music?
Yes <                    No

ould you like to select more than one variable to customize?
Yes <                    No
```

```
Which of these variables would you like to modify?
 Period <                  Composer                  Category <
 Theme                     Instrument <

Select the period to be featured during the block
 Early                     Baroque <                 Classic <
 Romantic                  Modern                    Contemporary

Select the Category you would like to feature:
 Symphony <                Concerto <                Ballet
 Tone Poem                 Vocal                     Chamber piece
 Solo                      Overture <                Recital
 Other
```

```
elect the instrument(s) you would like to feature in this block
 Piano  <                Strings                 Brass <
 Guitar                  Organ <                 Winds
 Harpsichord             Flute                   Clarinet
 Oboe                    Flute                   Cello
 Bass                    Violin

t time 01:00: 0, selection is CD-415104 #10 by HAYDN.
lassic,Mood=Med,Listen=High,size=Orchestral,Brass,cat=Concerto,14:0.
t time 01:14: 0, selection is CD-200011 #19 by PURCELL.
aroque,Mood=Soft,Listen=Med,size=Orchestral,Brass,cat=Concerto, 6:49.
t time 01:20:49, selection is CD-88187 #01 by VIVALDI.
aroque,Mood=Soft,Listen=Low,size=Ensemble,Organ,cat=Concerto, 7:0.
t time 01:27:49, selection is CD-412251 #04 by VON WEBER.
lassic,Mood=Med,Listen=Low,size=Orchestral,Piano,cat=Concerto, 18:0.
t time 01:45:49, selection is CD-55014 #07 by MOZART.
lassic,Mood=Med,Listen=Low,size=Chamber,Brass,cat=Concerto, 14:0.

Help     2Go     3WhatIf   4Variable 5Rule     6Set      7Edit     8Quit
```

LIST OF REFERENCES

Chorofas, D. N., Applying Expert Systems in Business, McGraw-Hill, 1987.

Cook, T. M. and Russell, R. A., Introduction to Management Science, Englewood Cliffs, New Jersey: Prentice-Hall, 1989.

Harmon, P. and King, I., Expert Systems: Artificial Intelligence in Business, New York: John Wiley and Sons, 1985.

Hayati, A. and Parker, A., "Automating the VLSI Design Process Using Expert Systems and Silicon Compilation," Proceedings of the IEEE, vol 75, no.6, June 1987.

Hoffman, R., "A Survey of Methods for Eliciting the Knowledge of Experts," SIGART Newsletter, April 1989.

Liebowitz, J., Introduction to Expert Systems, New York: Mitchell Publishing, 1988.

Mockler, R. J., Knowledged-Based Systems for Strategic Planning, Englewood Cliffs: Prentice-Hall, 1989.

Rolandi, W., "Knowledge Engineering in Practice," AI Expert, December, 1986.

Rolston, D. W., Principles of Artificial Intelligence and Expert Systems Development, McGraw-Hill, 1988.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Professor Magdi N. Kamel Department of Administrative Sciences Code AS/KA Naval Postgraduate School Monterey, California 93943	8
4.	Professor Hemant Bhargava Department of Administrative Sciences Code AS/BH Naval Postgraduate School Monterey, California 93943	1
5.	LT Ronald A. Boxall 3 Elm St. Holland Patent, New York 13354	2

Thesis
B78354 Boxall
c.1 SPEEDS.