NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

AN EXPERT SYSTEM APPLICATION FOR SERMIS

by

Daniel J. Rippinger

September 1988

Thesis Advisor: John B. Isett

Unclassified

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*): Code 54	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 Number; Project No; Task No; Work Unit Accession No
11 Title (*Include Security Classification*): An Expert System Application for SERMIS	
12 Personal Author(s): Rippinger, Daniel J.	
13a Type of Report: Master's Thesis; 13b Time Covered: From To	14 Date of Report (*year, month,day*): 1988 September; 15 Page Count: 100
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, knowledge acquisition

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

The Support Equipment Resources Management Information System (SERMIS) is an extensive program that controls the Navy's aviation support equipment assets. Deficient support equipment allowances for maintenance activities are being generated under this system due to the retirement of a cadre of experts who have acquired years of direct experience with these activities. These direct experiences contribute to mastering the types of knowledge SERMIS requires as input data to this allowance process. Interviews were conducted with an acknowledged expert to identify the types of information that comprised this prerequisite body of knowledge. A prototype expert system application was then constructed using a commercial development tool. The prototype demonstrated the feasibility of capturing this knowledge and using it as a front-end application to gain a more effective and efficient use of SERMIS.

20 Distribution/Availability of Abstract	21 Abstract Security Classification
[X] unclassified/unlimited [ ] same as report [ ] DTIC users	Unclassified

22a Name of Responsible Individual	22b Telephone (*Include Area code*)	22c Office Symbol
John B. Isett, Major USAF	(408) 646-2464	54Is

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.

An Expert System Application for SERMIS

by

Daniel J. Rippinger
Lieutenant Commander, United States Navy
B.A., Northwestern University, 1976

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
September 1988

Author: _____
Daniel J. Rippinger

Approved by: _____
John B. Isett, Thesis Advisor

_____
Carl R. Jones, Second Reader

_____
Dr. David R. Whipple, Chairman,
Department of Administrative Sciences

_____
Kneale T. Marshall, Dean of
Information and Policy Sciences

ABSTRACT

The Support Equipment Resources Management Information System (SERMIS) is an extensive program that controls the Navy's aviation support equipment assets. Deficient support equipment allowances for maintenance activities are being generated under this system due to the retirement of a cadre of experts who have acquired years of direct experience with these activities. These direct experiences contribute to mastering the types of knowledge SERMIS requires as input data to this allowance process. Interviews were conducted with an acknowledged expert to identify the types of information that comprised this prerequisite body of knowledge. A prototype expert system application was then constructed using a commercial development tool. The prototype demonstrated the feasibility of capturing this knowledge and using it as a front-end application to gain a more effective and efficient use of SERMIS.

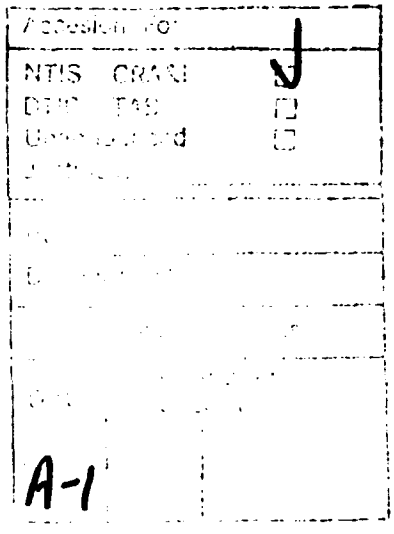

TABLE OF CONTENTS

ACKNOWLEDGMENTS

I would like to take this opportunity to acknowledge the people whose help was instrumental in the completion of this research. A sincere appreciation is extended to: the thesis sponsor for providing the opportunity and tools to become initiated in the art of expert system development; the expert, for contributing his experience, time, and most importantly, patience during the course of the interview sessions; my thesis advisor, for his insights and soft spoken insistence on producing a quality product; and finally my wife, whose emotional support, love and understanding were ever present throughout the entire process.

I. INTRODUCTION

A. BACKGROUND

Modern mission requirements of the U.S. Navy necessitate a variety of aircraft types. The Naval Aviation Maintenance Organization (NAMO) supports this diverse aircraft inventory by providing three levels of aircraft maintenance: organizational, intermediate, and depot. Organizational maintenance is performed at the squadron level and typically consists of the day-to-day repairs required on the aircraft. Intermediate maintenance activities (IMA's) are the next step in the maintenance hierarchy and they support a group of organizational activities that are usually in close geographical proximity. As an example, an IMA at an air station or on an aircraft carrier will support the various squadrons (organizational activities) based at that air station or assigned to the carrier. This level focuses on the upkeep of major aircraft components that are beyond the scope of an organizational activity, for example, engine overhauls, and radar calibrations. The depot level is a major re-work facility for the entire airframe. This type of activity conducts complete overhauls and performs major airframe modifications such as re-winging an aircraft. These three maintenance levels are summarized in Table 1.

Just as the complexity of maintenance increases with each level, so does the complexity of support equipment. The support equipment program needed to maintain these three maintenance levels has evolved into an extensive and complex program.

Technological advances required more varieties of support equipment. As the need for wider ranges of equipment grew, so did the requirement for an automated method to track the respective allowances, inventories, readiness reports, etc., for the various aircraft maintenance activities. A determination such as an allowance requirement for a specific

aircraft maintenance activity, would require consulting numerous types of ground support equipment (GSE) allowance lists. Manual determinations for these requirements became too time consuming and error prone. The Support Equipment Resources Management Information System (SERMIS) was created in response to this requirement for an automated system. The purpose of SERMIS is to provide management personnel with timely, accurate, and readily available support equipment allowance and data on this $6-$9 billion inventory. It maintains the necessary data for the effective management of aircraft support equipment. Appendix A provides a detailed description of SERMIS and its history.

TABLE 1. Aircraft Maintenance Level Summary.

Maintenace Level	Type Facility	Type Maintenance
Organizational	Squadron	Day-to-day operations and servicing
Intermediate	Centrally located for support of units on a particular base or ship	Repair, test, and modifications of aircraft components
Depot	Industrial	Rework and overhaul of aircraft, components and equipment

B. REASONS FOR THESIS RESEARCH IN THIS AREA

Even though SERMIS assimilates and aggregates the data in response to user inquiries, the inherent complexities of the support equipment system still exist. An example of such a situation was described in an interview (Appendix B) with the thesis

sponsor from NAMO concerning the Individual Material Readiness List (IMRL). The IMRL is a consolidated allowance list specifying support equipment end items and quantities required for an activity to perform its maintenance responsibilities. A typical intermediate level facility will have an IMRL list that is over 3,000 pages in length. SERMIS provides the data necessary for an accurate IMRL. There are training programs in place that teach IMRL managers the intricacies of navigating through SERMIS in order to obtain accurate IMRL's.

The problem then, is not inadequate training, but rather the experience level associated with the position of IMRL manager. There is an IMRL manager associated with each Support Equipment Controlling Authority (SECA). Presently there are six SECAs and the IMRL manager is responsible for the IMRL generation of each of the activities within the SECA. As an example, Commander Naval Air Force Pacific (COMNAVAIRPAC) is a SECA with over 300 activities.

The civilian holding the position of IMRL manager does not have any direct experience with these maintenance activities. It takes a substantial amount of experience to become familiar with all these activities. As an example, a new IMRL manager might be generating an IMRL for a particular air station. This manager is unaware that this station is required to support F/A-18's on a transient basis. Consequently support equipment for F/A-18's are left out of the IMRL. This error is not discovered until the IMRL is generated and goes to the activity. When the error is detected the IMRL has to be re-generated. All these facts regarding the intricacies of a maintenance activity are known collectively as the "employment data" for that activity.

When this IMRL manager finally departs the position, so will all the corporate knowledge on employment data obtained while using the system. Without this knowledge, maintenance activities risk the potential of operating with less than optimal levels of support

equipment. Therefore the problem becomes one of retaining the SERMIS knowledge in that position. There are other problems that contribute to an ineffective use of this system, such as a database that is not sound, available computer time, and so forth. The objective of the thesis however will focus on the capture and retention of the knowledge necessary for effective use of SERMIS. This will be accomplished through the use of an expert system.

The specific example of an IMRL manager was used to illustrate one of the problems arising from the complexity of SERMIS. Parallel problems can be found in any of the six functional subsystems (Appendix A) of SERMIS. For the purposes of this study, expert system construction will focus on the "allowance" functional subsystem which contains the programs necessary to compute support equipment allowances for various maintenance activities. These allowances are computed from established algorithms. This functional subsystem also has the capability to add to or modify these support equipment allowances.

C. HOW AN EXPERT SYSTEM ALLEVIATES THE PROBLEM

The sponsor has accomplished some initial work in this area through use of a commercial software package, "VP Expert" (see Appendix D for VP Expert details). The work thus far has been limited to applications that capture "textbook" knowledge, such as explaining SERMIS definitions for the user, presenting instructions, or indicating what references should be checked in order to obtain correct values for required data elements. The sponsor is now prepared to support further efforts in creating the necessary knowledge base.

Continued development is critical due to the decreasing supply of experts. At present, there are only a handful of people remaining that have been with the SERMIS program from its inception and that have had significant experience with the support equipment environment. It is this type of expertise that allows one to proceed correctly and efficiently

through a major SERMIS transaction. An example of such a transaction would be the determination of a new IMRL for an entire carrier airwing. This involves identifying the organizational and intermediate level support requirements for seven different aircraft types that contribute to the 90 plus aircraft total on board a standard aircraft carrier. It results in the 3,000 page IMRL output described previously. This type of transaction takes a weekend of computing time (Friday-Monday). Clearly, this is not a resource that can be used repetitively until one arrives at an optimal solution. The user has to be well versed in the ways of SERMIS in order to obtain the best results. It is not a matter of simply following menu choices.

Currently, this cadre of "experts" is not sufficient to fill all positions responsible for these types of major transactions. If such a calculation is required, and the expertise is not available in-house, commands have to "borrow" the proficiency from another location. This method obviously cannot suffice for the long term. An expert system, which would contain the acquired knowledge from a SERMIS expert, could be incorporated as a front-end system to SERMIS and will lead to a more efficient use of SERMIS without stretching the present limited resources.

D. RESEARCH QUESTIONS

The primary area that will be investigated is identification of the unique knowledge requirements of the SERMIS Allowance subsystem. What knowledge must be captured for the user to make more effective use of this subsystem? The SERMIS experts have acquired a certain knowledge level that distinguishes them from the novice user. What are the characteristics of this knowledge level? Is it composed of textbook knowledge or know-how gained through constant exposure to the system, or is it a combination of both? Once these essentials have been identified the research emphasis will be on prototype construction of an expert system in order to capture this knowledge. Can a simple expert

system be constructed using a commercial package? Does a commercial package provide effective knowledge acquisition? Finally, the matter of incorporation of an expert system into an established management information will be addressed. Should the expert system be integrated as part of the information system or should it be a separate component?

E. METHODOLOGY

Prior to undertaking any type of knowledge extraction interviews, a fundamental grasp of SERMIS had to be developed. This was accomplished in part by the sponsor interview (Appendix B) and by obtaining a copy of the SERMIS users manual. Study of the existing system was limited to the "textbook" exposure provided by the user's manual due to the inaccessibility of SERMIS sites in the area. The intent was to gain as much SERMIS domain knowledge as possible in order to reduce the communication overhead when it came time for the expert interview.

Once the fundamental concepts and terminology were acquired from the manuals, the expert interviews were conducted in order to transfer the knowledge from the human source to a knowledge base. During these interviews, the problem domain had to be restricted in order to keep the process manageable. For this study, the problem domain was limited to specific applications in the SERMIS Allowance subsystem. Concepts, terms, relations, definitions and basic strategies were uncovered that were relevant to these applications.

Following completion of the interviews, analysis and design of the expert system began using a prototyping approach. An initial knowledge base was constructed from the strategies revealed in the interviews through the use of the commercial software tool, "VP Expert." The rules comprising this knowledge base were then subjected to testing by the knowledge engineer. This was the basic iteration cycle used for the prototype

development. The main intent was to implement early and not become mired in making the informal rules perfect.

F. ASSUMPTIONS AND LIMITATIONS OF STUDY

One of the preliminary decisions to be made in developing an expert system is which software tool to use. An assumption of this study is that the software tool has already been decided as "VP Expert." This is the package that has been used in the initial cases performed by NAMO and is considered a suitable program.

Travel logistics were one of the major limitations of this study. The three main players in this study: sponsor, knowledge engineer, expert; were geographically dispersed between Washington, D.C., Monterey, CA, and San Diego, CA, respectively. This made coordination efforts between the parties somewhat difficult. The knowledge engineer's academic agenda also prohibited the scheduling of a series of dedicated one week intervals to conduct expert interviews. The expert interviews were conducted over a quarter break. This was the only available time period that contained a week that would not impact the academic schedule. This compressed window for knowledge acquisition meant limiting the scope of the knowledge domain to be extracted to a single application within the Allowance subsystem. The application chosen was the computation of allowance lists for fleet maintenance activities.

It is assumed that the usefulness of this effort will be a more efficient application of the SERMIS Allowance subsystem and computing time. This research will also lead to expert system development in other subsystems of SERMIS.

A final assumption of this research is that the recommended experts employed in the study had sufficient SERMIS experience to be considered "experts."

G. BENEFITS OF THE STUDY

The main benefit of this study was the result of developing the prototype expert system. This prototype established an initial, machine-usable knowledge base that provided a better understanding of the SERMIS Allowance subsystem. This increased understanding of SERMIS will ultimately lead to more efficient and accurate computations of support equipment levels to the fleet. The prototype will also serve as a foundation for further refinements and iterations to this subsystem and as a model for the other subsystems.

Expert system benefits can be generalized to many other areas of a military environment. A fact of life with military assignments is the "passdown" information associated with a job when it is turned over to one's successor. This passdown document is a repository of corporate knowledge that is manually maintained and updated. As with any form of manual documentation, it can eventually become too unwieldy and fall into disuse. An expert system offers an automated method of knowledge retention, and a more convenient manner to access this knowledge. The manual drudgery often associated with passdowns is now eliminated. Learning curves are reduced and a more disciplined understanding of the job is acquired.

H. ORGANIZATION OF THESIS

The remainder of this thesis is presented in three chapters. Chapter II discusses expert system theory and evolution and where this research falls in a general framework of expert systems. The elements of planning and building an expert system application are also covered. Chapter III details the specifics involved in the actual composition of the prototype expert system for SERMIS. This chapter provides descriptions of the expert interviews, and incorporating the extracted knowledge into a knowledge base. Chapter IV

presents the findings, conclusions and the requirements for future refinements. An extrapolation of expert system benefits to other areas is also presented.

II. THEORY OF EXPERT SYSTEMS AND KNOWLEDGE ACQUISITION

A. FRAMEWORK FOR EXPERT SYSTEMS

Expert system technology has, in recent years, exploded from the research environment into myriad real world applications. This rapid emergence has caused a "high-tech hype" to be associated with the term "expert system." When a technology such as expert systems, proves its potential and becomes accessible, companies are quite anxious to wrap their products in the cloak of this latest advancement. An aura becomes associated with the technology and suddenly it becomes a trend [Mishkoff 86]. Even though this aura has resulted in a higher visibility for expert systems, it has also created a more casual use of the term. Ultimately, a wide spectrum of products end up with the label "expert system." Some applications are deserving of the label and some merely are using it to remain in vogue. The original concept has been spread too thin across this range of applications.

1. Expert System Definition

A definition for "expert system" must be established as an initial step prior to any actual development.

A logical source would be one of the "founding fathers" of expert systems, Professor Edward Feigenbaum of Stanford University. Feigenbaum was a principal researcher for what is generally considered the first successful expert system application (DENDRAL). He defines an expert system as:

> ...an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution.
>
> The knowledge of an expert system consists of facts and heuristics. The "facts" constitute a body of information that is widely shared, publicly available, and generally agreed upon by experts in a field. The "heuristics" are mostly private, little-discussed rules of good judgement (rules of plausible reasoning, rules of good guessing) that characterize expert-level decision making in the field. The performance level of an expert system is primarily a function of the size and quality of a knowledge base it possesses [Harmon 85:p.5].

Boose describes an expert system as "a reasoning system that captures and replicates the problem-solving ability of human experts [Boose 86:p.2]." Other authors echo a similar view: that expert systems are programs containing expert knowledge; and they attempt to mimic an expert's process of making judgements in specialized areas [Ford 85, Hu 87:p.3, Michie 82].

For the purposes of this study, the definition of an expert system will adhere to the general flavor of these definitions, that is, an expert system will be considered a computer based program that manipulates a stored body of knowledge (derived from experts and facts) in order to arrive at a decision. Specific characteristics of an expert system will be addressed later in this chapter.

2. Evolution of Expert Systems

Man has constantly searched to enhance human abilities through mechanical assistance. The Industrial Revolution of the nineteenth century was a major outcome of these unremitting endeavors, while the development of the electronic digital computer is a principal consequence of these actions in the twentieth century.

Within this context of computer development, the objective was to develop a system which would replicate the human problem solving process [Harmon 85:p.2]. To accomplish this goal, two distinct and separate fields of study had to merge: psychology and computer technology.

Psychologists started placing a heavy emphasis on cognitive styles as a means of interpreting the human problem-solving process. "It [cognitive style paradigm] categorizes individual habits and strategies at a fairly broad level and essentially views problem-solving behavior as a personality variable [Keen 78]." The ultimate goal was to reduce this human problem-solving process down to a set of reasoning laws that the computer scientists could then encode into a program. It was discovered however, that the spectrum of human problem solving knowledge was too diverse to encode into a basic set of rules. The rules were simply too extensive for practical computer implementation. Reducing the set of rules limited applications to simple problems.

Time and technology were beginning to provide a solution however. From technology came hardware advances that permitted faster processing times and larger memory capacities. Roles for the computer were also becoming something more than just numerical computation tools. These roles were evolving from the strictly computational, to electronic data processing (EDP), to management information systems (MIS), to decision support systems (DSS)[Sprague 82].

The other part of the solution resulted from Artificial Intelligence (AI) researchers who concluded that knowledge capture had to be limited to specific domains for effective knowledge base construction [Buchanan 82]. Attempting to capture all of the human problem solving expertise was too formidable a task. If the knowledge could be relegated to narrow problem domains, then it could be comprehensively encoded. Figure 1, adapted

from [Harmon 85:p.3], reflects the merging of these two fields of study that have produced the concept of expert systems.

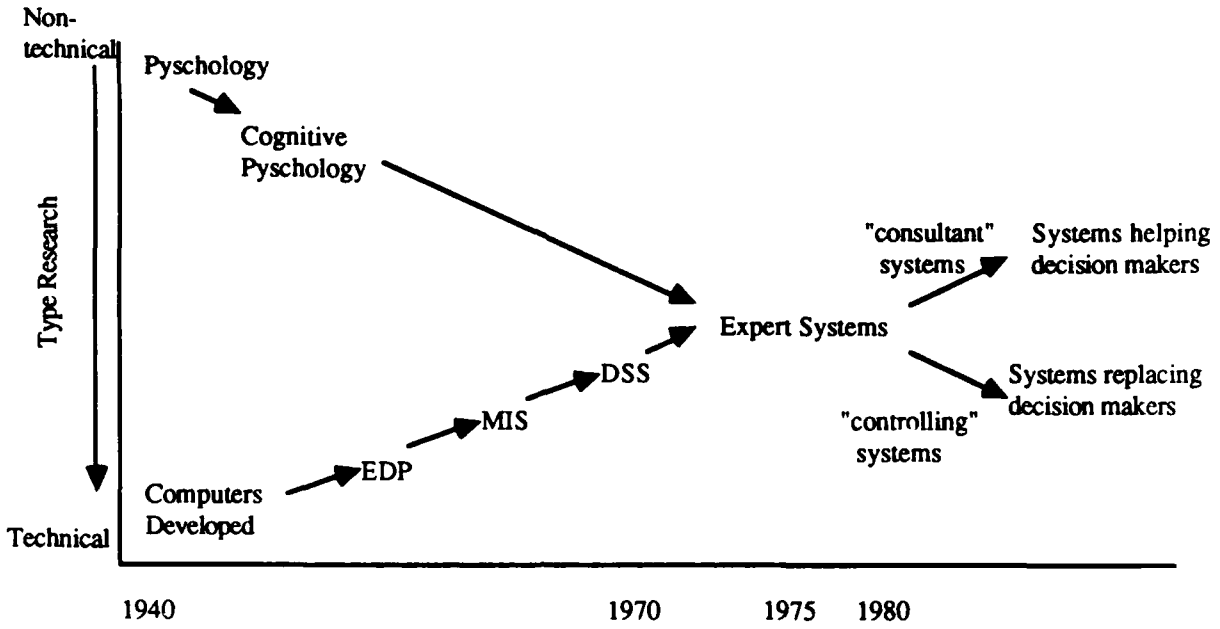

Figure 1. Evolution of Expert Systems

The term, "expert system" was unfolding as the embodiment of these attempts to capture human expertise in a specified area. Expert system applications were still for the most part limited to laboratory research until DENDRAL came on the scene, which is generally regarded as the first successful expert system application [Michie 82:p.3]. DENDRAL was a program that allowed a user to identify the structure of a complex organic compound based on an analysis of the compound's spectrogram. DENDRAL essentially proved that an expert's knowledge could be represented as a set of rules within a computer program. This program could then be used by someone less knowledgeable to achieve results similar to the expert.

Today expert systems serve mainly as intelligent consultants to decision making. The future will hold a time when these systems will actually take the place of the decision maker in certain areas [Ow 87].

3. Expert System Characteristics

Even though expert systems can vary significantly in terms of the type of problem that is being solved (see "Types of Expert Systems" section), all expert systems should reflect basic characteristics that are representative of the real life experts after whom the systems are being patterned.

Hayes-Roth et al. describe four basic attributes that are indicative of expert systems: quality, speed, specialized domains and an explanation facility [Hayes-Roth 83:p.42].

The importance of quality is obvious, that is, no one will want a system that produces inaccurate judgements. Speed is an important but relative feature of an expert system. Its weight will be dependent on the type of task. For example, if the task of a particular system is to provide control decisions at a nuclear power plant, system speed will carry an important emphasis. Caution must be taken however not to sacrifice quality for speed and vice versa.

Hayes-Roth et al. also state that experts are found only in "narrow, specialized domains [Hayes-Roth 83:p.42]." This is due to the fact that one can either possesses a good deal of knowledge (thereby considered an expert) regarding a limited number of subjects, or limited knowledge regarding a large number of subjects. In other words, if one "specializes" in a certain field, then the potential exists for becoming an expert in that field.

The final characteristic, an explanation facility, although not critical to the functionality of the system, does serve to satisfy the human operator's intuition and is

critical to the "consultant" systems. Little faith would be put in such a system that operated on a "black box" principle, that is, the inputs are put in at one end of the box and the answer magically appears out the other end. If a person can "see" the system understanding and working the problem, more confidence will be put in the results as well as gaining a better comprehension for the entire process.

Though all expert systems have these similar characteristics specific expert systems vary greatly however, due in large part to their tasks. Emphasis will now turn to an examination of these differences.

4. Types of Expert Systems

Expert system applications cover a wide range of tasks. The majority of expert system tasks come under a "consultant" label. Such tasks would include: debugging, design, interpretation, prediction, and diagnosis. The other generic and more advanced class of expert system types are "controlling" systems, which perform such tasks as: control, instruction, repair and monitoring [Hayes-Roth 83:p.13, Hu 87:p.186]. All of these tasks are explained below and summarized in Table 2, which has been adapted from the categories as described by Hayes-Roth et al. and Hu.

- Control systems analyze the current situation and anticipate future problems. The system will devise a plan to address the predicted trouble spots and then monitor the execution of the plan.
- Debugging systems when provided a malfunction, will recommend a remedy for that malfunction.
- Design systems take inputted constraints and arrive at a configuration that satisfy these constraints. Design systems provide an object configuration. A typical application may be the design of a computer network based on cost and size limitations.
- Diagnosis systems act as troubleshooters. Given the "symptoms" of a particular problem, the program will provide what type of malfunction is involved.
- Instruction expert systems combine diagnosis and debugging systems. Through use of the diagnosis subsystem, a student's deficiencies, that is, "malfunction" are identified. The debugging portion will address these deficiencies by presenting appropriate drills in order to correct them.

- Interpretation systems analyzes incoming data and assigns it a "real world" meaning. It deciphers observed data. A typical example in this area would be an expert system that takes instrument readings in order to recognize deviations.
- Monitoring systems are continually comparing system observations to the the system goal or factors critical to a successful outcome. If a discrepancy is noted, the monitoring systems can then identify a situation that would conflict with planned outcomes.
- Planning systems provide a series of actions to take to achieve a goal. The plan also has to meet any constraints that are given. One such example would be a plan to test an installed computer system.
- A prediction type of expert system forecasts future outcomes. The prediction is based on knowledge of events and circumstances that caused past occurrences of similar outcomes.
- Repair systems take the combination of debugging and planning systems to their logical conclusion, that is, they fix a problem once it has been diagnosed. A problem is identified, a plan is made to correct the problem, and the plan is then executed.

TABLE 2. Categories of Expert Systems' Tasks

Category	Problem Addressed	Sample Application
Control	Constantly analyzes, predicts & repairs system behavior	Air traffic control
Debugging	Prescribing remedies for malfunctions.	Treat nuclear reactor accidents.
Design	Configuring objects subject to a set of constraints	Computer network
Diagnosis	Inferring system malfunctions from "symptoms"	Diagnose diseases
Instruction	Diagnosing, debugging, and correcting student behavior	Instructing naval students in steam plant operation.
Interpretation	Inferring situation description from data interpretation	Interpreting sonar sensor data
Monitoring	Comparing observations to goals/vunerabilities	Fiscal management
Planning	Designing actions to achieve given goals	Test plans.
Prediction	Forecasting consequences of given actions	Military forecast of where next armed conflict would occur
Repair	Executing a plan to administer a prescribed remedy	Repair of automotive components

5. Expert Systems vs. Other Automated Technologies

An expert system has been defined as a program that manipulates a captured body of domain specific knowledge in an attempt to mimic a human expert's problem-solving process. Even given this generally accepted definition, the term "expert system" is at times, confused with terms from related, but distinct automated technologies. Additional clarification is needed to firmly establish expert systems in this framework of automated technologies.

a. Expert Systems and Decision Support Systems

The first differentiation that should be addressed is the one between expert systems and DSS. Ford states that the differences between these two are in four areas: objectives, operational use, users, and development methodology [Ford 85:pp.24-25].

With respect to objectives, Ford contends that DSS "support the user" in making a decision while expert systems "provide to the user a conclusion or decision [Ford 85:p.24]." Sprague and Carlson corroborate this passive role of DSS by their definition which states DSS "help decision makers confront ill-structured problems [Sprague 80:p.1]." The difference here is that expert systems take a more active role by *providing* a solution or conclusion to the user while DSS *supports* the user by making data and models critical to the decision more accessible and quantifiable.

Operationally, DSS are more flexible, that is, they permit the user to "manipulate the data and models in a variety of ways while progressing through the decision making process [Ford 85:p.24]." Expert systems are much more regulated. The knowledge base of an expert system generally consists of a set of rules. The conclusion is reached by a system, not user, manipulation of these rules. This manipulation is accomplished through the use on an inference capability [Harmon 85:p.34]. Consequently,

the operational difference with DSS, is the user directs the system, and with expert systems, the system directs the user [Turban 86:p.122].

The next difference is found in the type of users. In the beginning days of expert systems, the difference between DSS users and expert system users could be classified as business vs. research [Ford 85:p.25]. This particular difference is becoming less clear cut over time however as expert systems are gradually finding a niche in business organizations. DSS are designed for improving the effectiveness of individual decision making within a business organization. Since DSS applications can cover the concerns of a diverse group, these users are normally considered a non-homogeneous group. In contrast, expert systems were developed from a narrow knowledge domain and therefore the users are a more homogeneous group. Ford makes a further distinction by stating that DSS users normally helped designed the system, while most expert system users had nothing to do with the design [Ford 85:p.25].

The final area of contrast concerns the development methodology. Both DSS and expert systems employ the prototyping approach. For DSS this approach permits a high level of user involvement and allows the application to continue to adapt to the user's needs. When this iterative approach is used in the development of expert systems, it allows the knowledge base to be constantly refined and expanded, which leads to better performance [Ford 85:p.25].

b. Expert Systems and Artificial Intelligence (AI)

Finally, a distinction should be made between expert systems and the more inclusive category of AI, since there exists an occasional propensity for these terms to be used interchangeably.

AI is generally considered an outgrowth of computer science that deals with the development of systems which attempt to produce results on a level equal with human

intelligence [Harmon 85:p.3]. Expert systems are just one branch of AI. Other AI categories include: robotics, voice recognition and synthesis, and vision [Hu 87:p.3]. Establishing a precise definition of artificial intelligence is a very elusive task. Since the field is constantly changing, so are the definitions. The intent here is not to nail down the definition of artificial intelligence, but rather to establish that expert systems are normally considered a subset of this fluid technology .

6. How SERMIS Project Fits in the Framework.

Various characteristics and types of expert systems have been described. In general, expert systems adhere to the same basic architecture (see "Expert System Components" section) but differ in their knowledge bases and applications [Hayes-Roth 83:p.89].

Luconi et al. provide a framework that puts expert systems into a management context [Luconi 86:p.6]. This framework is depicted in Figure 2. The problem types used by this framework are defined as:

- Type I: where the problem elements of data, procedures, goals & constraints, are all clearly defined and therefore suited for conventional programming techniques. This type of problem is fully structured and is indicative of data processing.
- Type II: an unstructured quality to the problem elements is introduced. Standard procedures alone are no longer sufficient. Computers apply procedures to the structured component but rely on humans to decide which procedures are appropriate in a given situation. Decision support systems deal with Type II problems.
- Type III: the system is capable of encoding some of the goals and strategies from humans. The system can now explore a range of strategies before picking a solution. Luconi et al. label this type problem appropriate for expert systems. They argue however, that it appears to be almost impossible to encode all the knowledge for less clearly bounded problems like financial analysis [Luconi 86:p.9].
- Type IV: all the problem solving knowledge cannot be feasibly encoded, but it is still beneficial to draw on expert system techniques. Luconi et al. state this type of problem is well suited for what they label "Expert Support Systems." These systems focus on a design that supports rather than replaces users [Luconi 86:p.9].

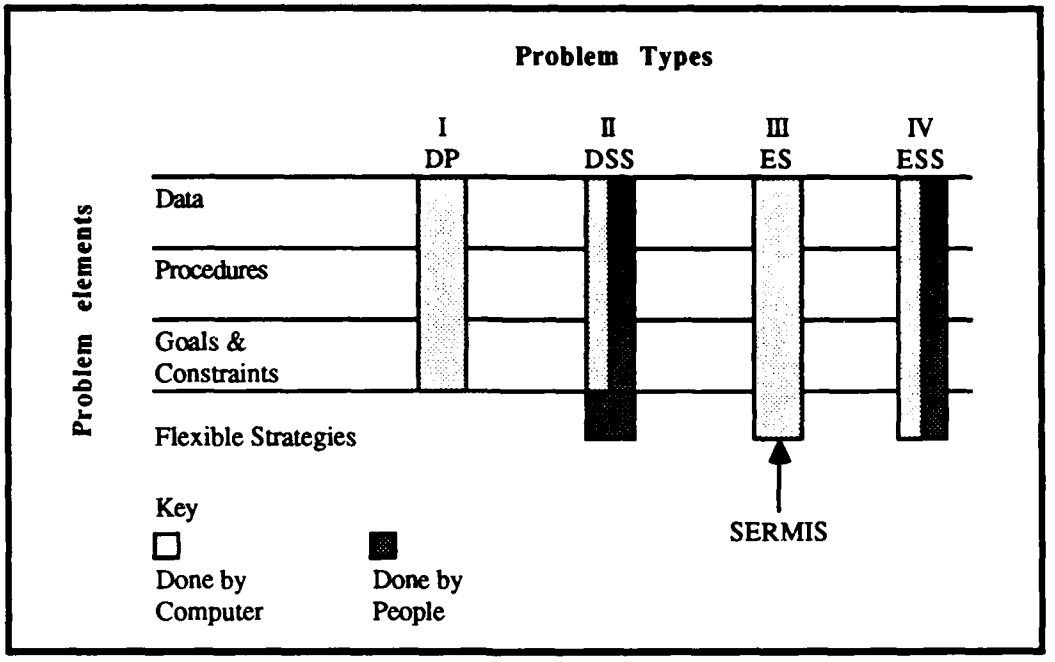

Figure 2. Expert Systems Framework [Luconi 86:p.12].

The problem elements are represented on the vertical axis. The type of problem will depend on how structured these problem elements are. The authors described this framework as an extension of the earlier framework of Gorry and Scott Morton [Luconi 86:p.6].

The SERMIS effort will be a rule based, planning type expert system that will fall under the Type III category. It is a Type III problem because it is a prototype effort, and consequently will only cover a limited area of SERMIS. Future endeavors may cover a wider spectrum placing these efforts in the Type IV category since there will be a larger amount of potentially relevant information.

Blanning states that a "reasonable approach", in regards to expert system design and implementation, "is to begin with partially structured but interesting problems and then move on to less structured ones as experience is gained [Blanning 84]." This makes the SERMIS effort a favorable candidate for an initial expert system construction.

B. EXPERT SYSTEM COMPONENTS

Regardless of type application, all expert systems share a basic set of components: knowledge base, inference engine, and user interface [Luconi 86:p.4]. These components are shown in Figure 3.

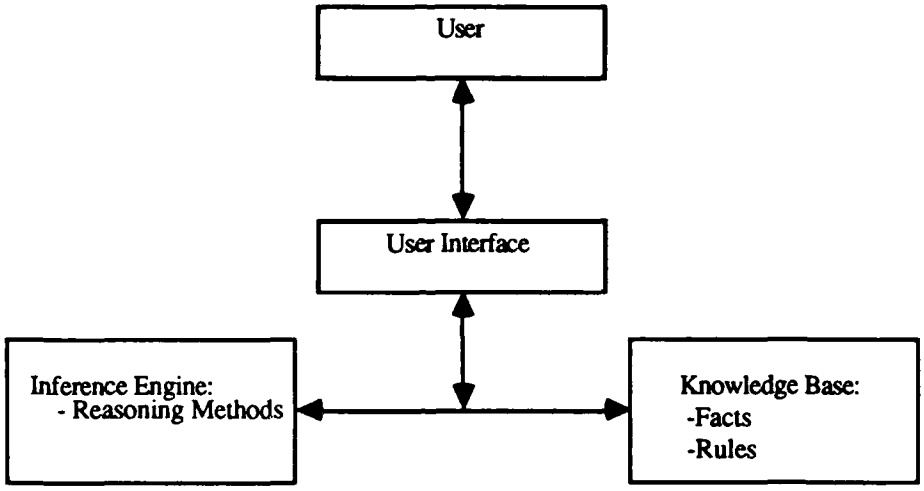

Figure 3. The Basic Components of an Expert System. [Luconi 86:p.7]

1. Knowledge Base

A knowledge base contains the encoded knowledge acquired from the expert. Prior to incorporation into the knowledge base however this knowledge can exist in many forms [Hayes-Roth 83].

a. Types of Knowledge

Learning is a result of knowledge acquisition from two different domains. Harmon and King label one of these domains as consisting of facts, and the other consisting of "heuristics [Harmon 85:p.30]." The acquisition of facts is accomplished through textbooks and other formal methods of education. Heuristics however are learned through experience. They are "rules of thumb" that one relies on to solve a problem or complete a task [Harmon 85:p.31]. Heuristics are the means by which the facts are applied

to tasks. For example, a law student may know a great deal of factual knowledge concerning laws and cases but until actual experience is gained in the courtroom the overall expertise will be limited.

Figure 4, from Harmon and King, illustrates the general pattern for compiling knowledge [Harmon 85:p.33]. The process begins with an integration of formal methods and experiences. But as time progresses it ultimately relies on knowledge gained through experiences. Feigenbaum also supports the belief that the nature of the knowledge an expert has acquired is "largely heuristic [Feigenbaum 84:p.53]." This trend of knowledge compilation over time is depicted by the large arrow in Figure 4.

2. Inference Engine

The inference engine is the link between the knowledge base and the working memory. It is the component that manipulates the knowledge base in order to achieve a solution. The inference engine performs this manipulation by deciding how to search the knowledge base, which rules or frames to employ, and when to stop and present an answer.

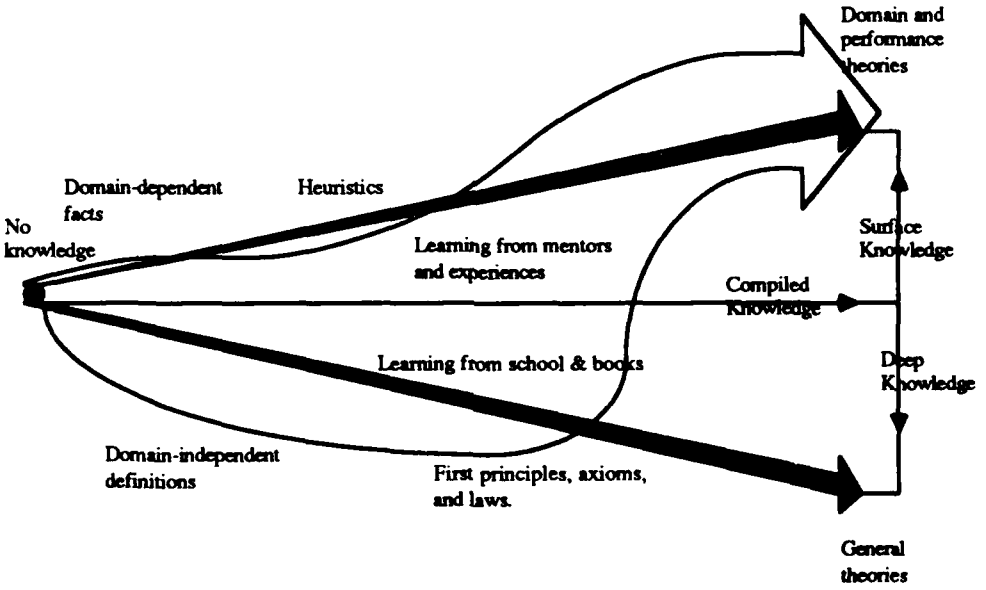

Figure 4. General Pattern of Professional Development [Harmon 85:p.33].

3. Interface

The interface allows the user to interact with the expert system. This interface can include a wide variety of features such as: explanation facilities, graphical representations, and on-line help functions. The interface should make expert system operation easy for the user and as such is a key factor for determining frequency of use [Harmon 85:pp.98, 205].

C. PLANNING AND BUILDING AN EXPERT SYSTEM

There are a variety of reasons for wanting to implement an expert system. This section examines these criteria, discusses how the knowledge is extracted and incorporated, and the methodology used in the construction of this prototype system.

1. Criteria

There are several reasons why an expert system should be constructed as opposed to continued reliance on human experts. Boose gives the following justifications [Boose 86:p.3]:

- Experts retire.
- Scarcity of expertise.
- Expert systems provide answers faster, if human expertise is not readily available.
- Expertise might be expensive to obtain.
- There are better ways to optimize an expert's time than to instruct users.
- Experts are not always consistent.

The first three reasons are the ones most germane to the SERMIS undertaking.

Additionally, Vedder and Mason argue that five criteria, which are more application specific, should be satisfied prior to considering a development. These criteria along with their applicability to SERMIS are depicted in Table 3 [Vedder 87:p.402]. SERMIS meets both criteria sets presented here.

TABLE 3. Application of Development Criteria for the Use of Expert Systems

General Criteria	Applicability to SERMIS
1. Numerous instances of the situation occur for which the choice of an effective response depends on a common body of knowledge and/or expertise.	IMRL generation has to be performed for every naval aviation organization.
2. Experts are available	There are personnel that manually generated IMRLs prior to SERMIS
3. But expertise is scarce and expensive to develop through education and training	There are not enough experts for each SECA. They are retired or approaching retirement.
4. The relevant knowledge in principle is articulable, that is, can be communicated to a knowledge engineer (and captured in an expert system)	Interviews with one expert provided some suggestions that can be employed.
5. Failure to apply this expertise can result in significant losses.	Erroneous use of SERMIS can result in the wrong type of support equipment for deploying units

2. Role of the Knowledge Engineer

One of the most instrumental functions in the development of any expert system, is the extraction of knowledge from the sources (books, experts, etc.) into the expert system. The person performing this task is known as the knowledge engineer. Figure 5 [Hayes-Roth 83:p.130], illustrates the relationship the knowledge engineer has between the expert system and the expert.

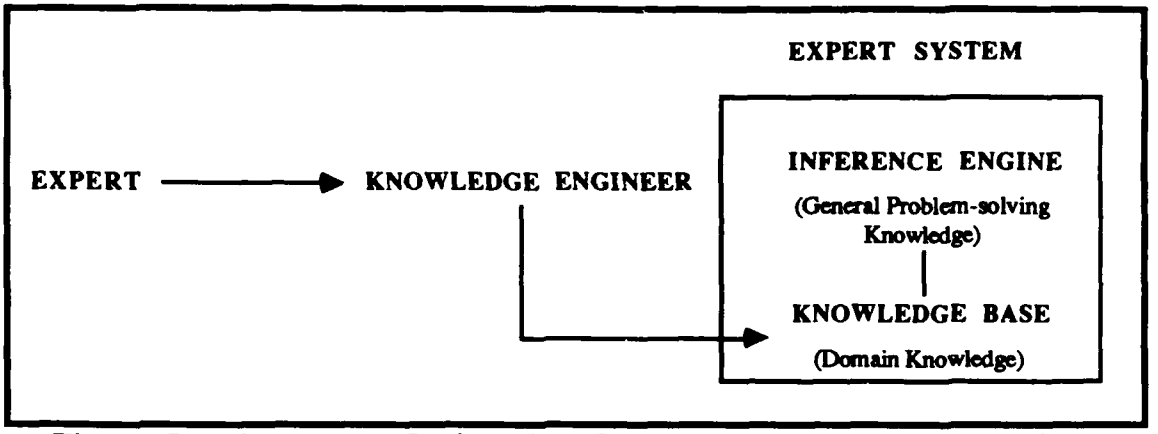

Figure 5. Knowledge Engineering--Expert to Knowledge Base Via a Knowledge Engineer

Olson and Reuter give two methods by which the knowledge engineer can acquire the knowledge: expert interviews and becoming the expert [Olson and Reuter 87]. Becoming the expert is not a feasible solution due to the tremendous amount of time that would be involved attempting to acquire the knowledge and then building the expert system. Yet this approach is required, to a limited degree in order to gain an operating knowledge of the problem domain. In other words the knowledge engineer has to learn some of the "textbook" portions of the knowledge in order to comfortably converse with the real expert during interviews. This will require becoming an "expert" with respect to certain fundamentals of the problem domain.

The major source of knowledge will then come from the domain expert [Feigenbaum 84, Waterman 86]. A knowledge engineer now must not only be technically competent, but also be proficient in a wide range of inter-personal skills as well. Some of these skills are: good communicator; tact and diplomacy; persistence; versatility and inventiveness; patience; and logicality [Hart 86:p.40]. All of these are critical in eliciting accurate knowledge from the expert by means of interviews. Sources state than effective expert interviews can be a "careful, painstaking analysis [Feigenbaum 84:p.53]" that last "over a period of many months [Waterman 86:p.152]." Because of all the skills, time and

complexities involved with the knowledge acquisition task, it is often considered the bottleneck in developing expert systems [Hayes-Roth 83, Heng 87, Olson 87].

Recent efforts to reduce this "bottleneck" have included tools that automate some of the knowledge engineer's tasks. Waterman breaks them into four major categories: programming languages, knowledge engineering languages, system-building aids, and support facilities [Waterman 86:p.80]. Summary descriptions are presented in Table 4.

TABLE 4. Summary of Expert System Tools.

CATEGORY	DESCRIPTION
Programming Languages	Languages designed for a specific class of problems
Knowledge Engineering Languages	Languages expressly for Expert System building
System-Building Aids	Programs that help acquire and represent expert's knowledge
Support Facilities	Programming tools, e.g., debugging aids, knowledge base editors

3. Knowledge Representation

It is understood that the knowledge base contains the expert's knowledge, but exactly how is this knowledge represented in a program? Some sources present as many as five strategies for encoding knowledge into the knowledge base [Harmon 85:p.35]. For the purposes of this study, only the three most widely employed methods: rule-based, frame-based, and semantic nets; will be discussed [Waterman 86:pp.63-79].

Rule-based systems capture knowledge by employing one or more logical "if-then" statements. The IF portion presents a condition, and when this condition is judged valid, the THEN statement provides an action to take. As an example, a simple rule used to determine what level of aircraft engine repair a particular facility provides might read : IF repair includes removal and replacement of compressor rotor, THEN maintenance activity

is a first-degree repair facility. The rule complexity can vary from one to many conditional statements.

Frames are another means of representing knowledge. They are used to denote common concepts and situations [Waterman 86]. Frames are more modular than rules since each frame contains information related to one subject. Information within the frames is broken down into two categories - slots and procedures. Slots are the attributes of the object or situation being described, and the procedures are executable code associated with each slot. The main difference with this method, as compared to rule-based systems, is that frames can have default values put into the slots. These values will then be used if other information is not provided.

A third method of knowledge representation is a semantic net. The net consists of nodes and arcs. Nodes are points that represent objects, and arcs are the links that connect the nodes. The arcs describe the relation between the nodes. Two of the more common arcs are " is a" and "has a" [Waterman 86]. Figure 6 is an example of a semantic net with four nodes and three links. A primary advantage of a semantic net is its ability to establish an inheritance hierarchy. This means that objects lower in the net hierarchy inherit properties from objects higher in the net. In the example from Figure 7, USS Nimitz would automatically inherit the property of "ship" from the object "aircraft carrier" , since "Nimitz" is lower in the hierarchy than "aircraft carrier." The inheritance hierarchy saves space by not having to duplicate information [Waterman 86].

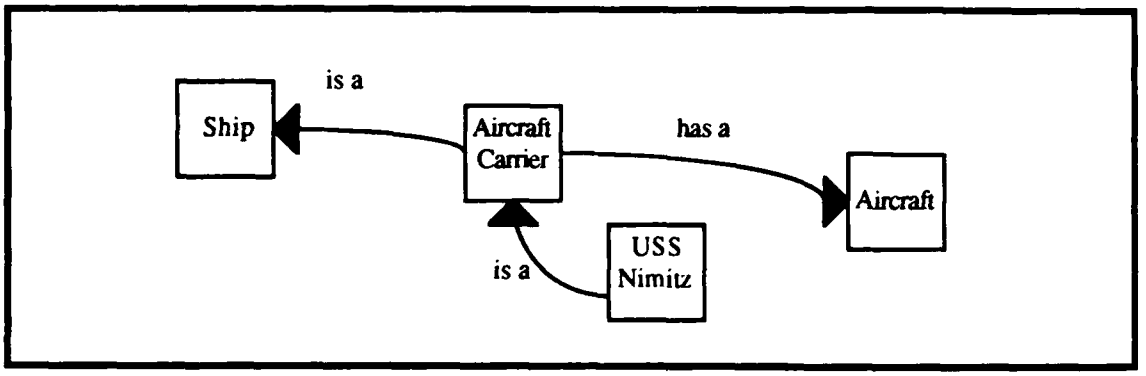

Figure 6. Example of a Semantic Net.

The three main methods of knowledge representation are summarized in Table 5.

TABLE 5. Primary Methods of Knowledge Representation.

METHOD	DESCRIPTION	ADVANTAGES	DISADVANTAGES
Rules	A set of rules are checked against the facts of the current situation. When a rule premise is valid, the "THEN" action is executed.	- Most popular - Flexible	- Some systems require a large quantity of rules
Frames	A body of related knowledge consisting of slots (attributes) and procedures	- Slots can provide default values.	-Harder to maintain
Semantic Net	A collection of nodes connected by links that represent the relationships between nodes.	- Inheritance hierarchy	-difficult to use for loosely established taxonomies

4. Knowledge Base Search Methods

Inference engines normally use one of two principal search methods when implementing the knowledge base: forward chaining, or backward chaining.

Backward chaining is known as a goal driven method because the inference engine starts with a goal and works backwards through the rule conclusions attempting to to determine if the premises are valid [Boose 86]. In order to determine a ship type (the

goal), for example, backward chaining would start with a specific ship type, an aircraft carrier for example, and then work backwards to determine if all the conditions are met. If it found an aircraft carrier condition that was not met, it would abort that choice and then assume another type ship, such as a frigate, until a ship type was found that met all the conditions. Backward chaining is an efficient methodology when there are a small number of goals compared to the number of initial states.

Forward chaining is said to be a data-driven system. In this case the goal needs to be assembled from a relatively large amount of possible outcomes [Harmon 85:p.55]. The system is given the known facts and then conclusions are drawn from the rule premises. The inference engine keeps working forward, adding any valid conclusions to the facts until there is enough information for a solution. For example, consider a knowledge base containing these two rules: 1) IF ship displaces more than 50,000 tons and embarks fixed-wing aircraft, THEN ship is an aircraft carrier; 2) IF ship is an aircraft carrier and hull number is 68, THEN ship is USS Nimitz. Now if the facts of "ship displacement is 90,000 tons" and "ship carries fixed-wing aircraft " are supplied to the knowledge base, the premise to the first rule is met and forward chaining will now add a third fact, that the ship is an aircraft carrier, to the knowledge base. This fact is now available for use in the second rule.

5. Development Methodology

When it comes to the development of expert systems, there may not be any "time-tested methodology [Heng 87:103]," but the consensus would appear to advocate the use of prototyping, since its success has been demonstrated with decision support system development [Ford 85, Henderson 87, Heng 87, Hogue 84, Waterman 86]. Although this methodology may be known by a wide variety of pseudonyms, such as: adaptive design [Alavi 84], evolutionary development [Waterman 86], generation-and-test [Feigenbaum

84], and rapid prototyping [Philips 88], the inherent advantages of this approach remain the same.

With prototyping, a basic working version of the system is implemented and tested as soon as possible. Portions of knowledge are extracted, coded, implemented and tested in incremental steps. The expert critiques that portion of the system, the knowledge engineer notes the deficient areas and produces the next version. The intent with this prototype is not to test the accuracy of code or speed of performance, but rather, the "adequacy of the formalization and of the basic underlying ideas [Waterman 86:p.146]." Often times a prototype will never serve as a final version because "it can no longer support the new knowledge added and has to be discarded [Heng 87:p.103]." But it has served its purpose if it has pointed out potential solutions for subsequent versions [Ow 87:p.446].

Expert system development shares the same benefits as decision support development when employing the prototyping methodology, that is it: provides an assessment of benefits [Keen 81], increases user involvement [Hogue 84], serves as vehicle to clarify user requirements, demonstrates project feasibility, and it can identify potential trouble spots early on.

Expert systems however, can gain further advantages due to some inherently unique characteristics of its own. Besides helping define system requirements, Heng presents three additional problem areas where the prototype approach is tailored for helping expert system developers: the problem of knowledge extraction; the problem of structuring the knowledge for machine manipulation; and the problem of maintaining the interest of the experts [Heng 87:p.103]. Prototypes are a fact of life when it comes to expert systems. The built in complexities of knowledge extraction, coding, and expert interviewing preclude carrying out this process in a linear fashion. "The prototype is a tool to cope with the evasive nature of human expertise [Heng 87:p.106]."

6. Validating an Expert System

An expert system which has not been continuously validated will provide low quality decisions which in turn leads to a loss of confidence in the system and in an eventual disuse of the system [O'Leary 87]. As discussed in the prototyping section, the expert system will be subjected to continual validation by the expert if the knowledge engineer is indeed getting feedback on each version. Feigenbaum considers this constant interaction between developer and expert essential. Based on his experiences:

> The expert was surprised, sometimes frustrated, by the occasional gaps and inconsistencies in the knowledge, and the incorrect diagnoses that were logical consequences of the existing rule set. The interplay between the knowledge engineer and expert gradually expanded the set of rules to remove most of these problems [Feigenbaum 84:p.51].

Quite often, the human specialist is rarely successful in conveying an accurate, objective description of their subjective thought processes to another person [Heng 87:p.106]. Even when the knowledge engineer thinks a firm grasp on the knowledge has been attained, it still may not be compatible with the expert's concept. This is why the knowledge must be continually tested and validated. It also represents why expert system testing has to incorporate more than just the traditional method of error checking the software [O'Leary 87].

D. INTEGRATION OF AN EXPERT SYSTEM INTO AN MIS

Turban and Watkins argue that most of the expert systems in use today are independent applications. Now that expert systems have proven their value in specialized fields, the time has arrived to start considering how this newer technology of expert systems can be incorporated into already established information systems [Turban 86]. This integration can be in either of two forms: an expert system(s) integrated into the

existing components of the information system; or an expert system as a separate component of the information system. Since expert systems deal with specialized domains, the method of integration is dependent on the scope of the existing management information system (MIS). Several expert systems may have to be integrated into an MIS that is broad in scope so that there is a separate one for each domain. On the other hand, an expert system as a separate, or front-end, component might be warranted if the information system has a narrow domain. [Turban 86]. Regardless of what form the integration takes, the main benefit gained is that both types of technologies have "distinct advantages that, when combined can yield synergetic results [Turban 86:p.123]."

An additional benefit to be gained from the integration of expert systems and existing MIS is the closing of the gap that exists between management expertise and computer competence. This discontinuity is unique to today's technological climate. Upper level managers have a wealth of managerial skills and knowledge by virtue of their experiences, yet they attained their positions during a time when computer applications were still in their infancy. The individuals most skilled in computer development however, can be characterized as a relatively young group with minimal managerial skills. Consequently, there are managers with minimal computer training being provided decision support software by specialists with minimal managerial skills. Managers therefore do not have complete confidence in these systems, which Cooper argues is a major reason for their disuse [Cooper 86]. Cooper goes on to suggest four criteria that a decision aiding tool must meet in order to close this gap: 1) require no user training; 2) provide an explanation facility; 3) require no customization; 4) Incorporate state-of-the-art techniques [Cooper 86:pp.220-221]. These criteria are presented in Table 6 along with the reasons why it can help bridge these differences. Expert systems meet these criteria and therefore should lead to a greater use of current automated information systems if they can be integrated.

TABLE 6. System Criteria Required to Close Managerial/Computer Competence Gap.

CRITERIA	REASON
Requires no training	Makes the system less intimidating
Has an explanation facility	Establishes a credibility
Requires no customization	Expert systems should learn from the user and not be "customized" in order to fit the user
Incorporate state-of-the-art analytic techniques	The techniques remain when the expert goes home

E. SUMMARY

Expert systems have evolved out of a merging of the psychology and computer disciplines. These systems "preserve and disseminate scarce expertise [Luconi 86:p.3]" by encoding facts and the experiences of an expert into a knowledge base. This knowledge base is then accessible to less experienced people through manipulation by an inference engine. The design and construction of expert systems has also resulted in the creation of a new speciality - the knowledge engineer. The knowledge engineer is charged with the representation, organization, transfer, utilization, and extension of the knowledge used by the expert system [Tou 85]. This task requires an unique blend of technical and personal skills. The prototyping or evolutionary approach is considered the most effective means of expert system design since it accommodates the "trial and error" method often associated with knowledge extraction. Once an expert system is developed, integration with an organization's current information systems can provide a greater synergy for decision making as well as bring the human expertise and judgement (managers) together with the power of computers.

This chapter has laid the groundwork for the SERMIS prototype effort. The subsequent chapters describe how the knowledge engineer extracted an expert's knowledge and incorporated this knowledge into a rule-based prototype expert system. This system can be used in a "consultant" role to aid SERMIS users at the Support Equipment Controlling Authority (see Appendix B) level in the generation of support equipment allowances to their cognizant activities.

III. DESIGN AND CONSTRUCTION OF A PROTOTYPE EXPERT SYSTEM FOR THE SERMIS ALLOWANCE SUBSYSTEM

The basic theory and concepts of expert systems have been addressed in Chapter II. These concepts were put into practice in order to produce the prototype application for the SERMIS Allowance subsystem. This prototype attempted to capture the expert knowledge necessary to produce accurate support equipment allowances for fleet maintenance activities. This chapter details the steps taken in the design and construction of this prototype.

A. INTRODUCTION

A knowledge engineer goes through several stages prior in producing an expert system. Hayes-Roth et al. has broken these stages of knowledge acquisition into five areas. These five stages and their relationships are shown in Figure 7. The development and construction of this prototype system subscribed to these general stages. Each stage will be addressed individually in this chapter.

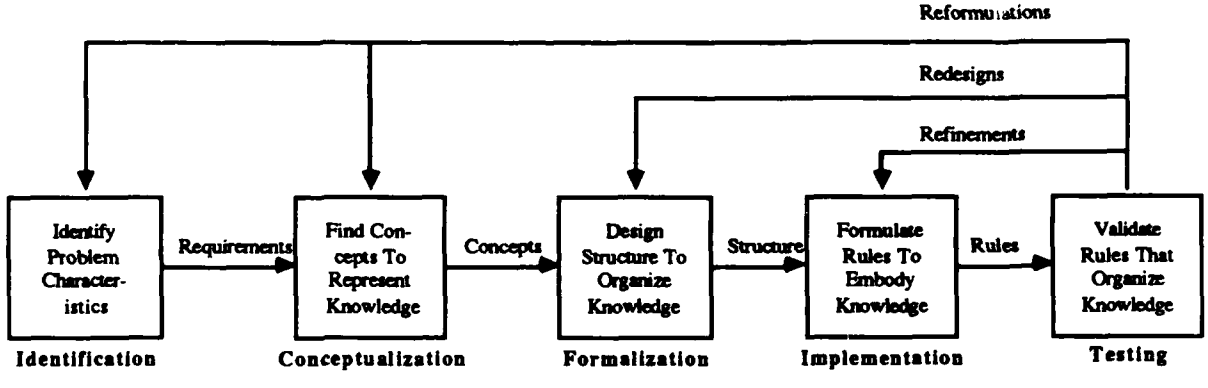

Figure 7. Stages of Knowledge Acquisition [Hayes-Roth 83:p.139].

B. IDENTIFICATION STAGE

This is the initial step in knowledge acquisition. The intent of this stage is to identify the players, problem characteristics and resources that will be involved in the development of the expert system [Hayes-Roth 83:p.141].

1. Players

The construction of an expert system is centered on four main players: the expert system, the domain expert, the knowledge engineer and the expert-system-building tool [Waterman 86:8]. For this project, the expert system was the final package--the inference engine along with the specific knowledge bases developed for this project. The expert was located at Commander Naval Air Forces Pacific (COMNAVAIRPAC) headquarters; the author served as knowledge engineer; and the commercial application--"VP-Expert" (Appendix D)--was used as the development tool to construct this first version prototype.

In order to meet with the expert, interviews were scheduled over a three day period at COMNAVAIRPAC headquarters in San Diego, California (see Appendix C for a summary of the interview period). Unfortunately, this was the sole opportunity to interact with the expert due to academic and funding constraints. Nonetheless, it did serve as a solid foundation for establishing the fundamental concepts.

2. Problem Identification

A very specific area of SERMIS had to be identified as a target application for this prototype expert system. This specificity was required in order to optimize the limited time available for interviews. Based on the example cited in the meeting between the thesis sponsor and author (Appendix B), it was decided that the work would focus on the IMRL generation process (done in the "Allowance" subsystem of SERMIS) for the activities within the COMNAVAIRPAC SECA. As a reminder--IMRL's are the authorized support equipment allowance for each aviation activity. These IMRL's are prepared for an activity by the cognizant SECA [NAVAIRINST 87]. Appendix A lists the six SECA's.

The first interview session was spent eliciting potential causes for inadequate IMRL generation. One of the sources identified were the civilians doing the inputs at the SECA level. The expert contends that these civilians have no real feel, experience or incentive to ensure all inputs regarding the IMRL activity are correct. The operator's actions are divorced from the consequences. The other major problem area singled out was the lack of data integrity in the SERMIS source data. This particular problem however, is the concern of the Naval Aviation Engineering Center (NAEC) and goes beyond the scope of this work.

Figure 8 depicts the problem sources that were identified as a result of the first interview session. The shaded region depicts the area that will be addressed by this research.

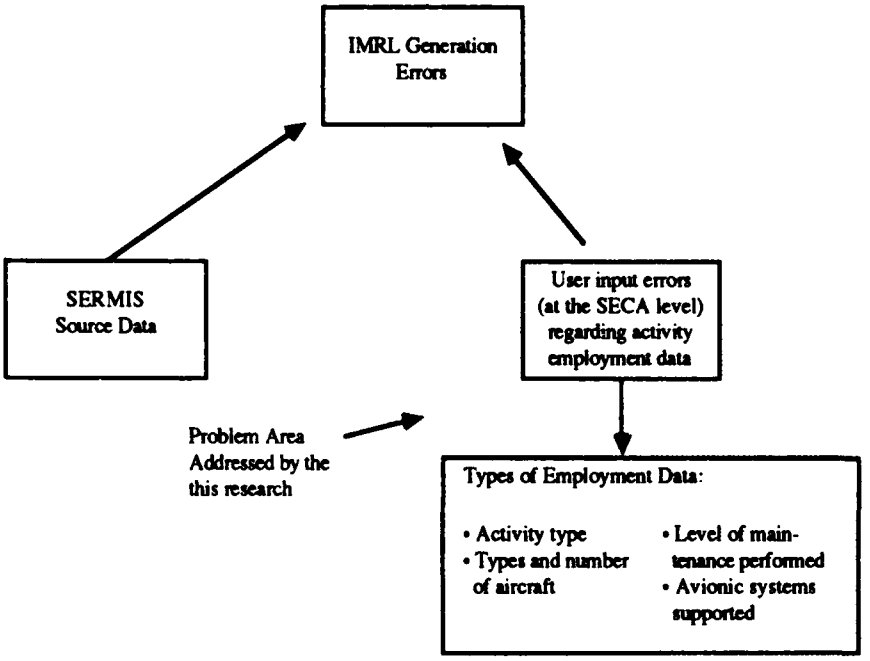

Figure 8. IMRL Generation Problem Areas.

In other words, this prototype application intends to improve on the problem of inadequate IMRL generation by making employment data "expertise" available to the civilian operators at the SECA level.

3. Resources

The four major resources required to accomplish the prototype construction included: knowledge sources, time, computing facilities and money [Hayes-Roth 84:p.142].

The knowledge sources included both the domain expert and textbook references. The thesis sponsor identified the expert, arranged for the interviews and provided a copy of the SERMIS user's manual. During the course of the interviews the expert described many background references, such as NAVAIR instructions, various excerpts and maintenance manuals that were previously unavailable to the knowledge engineer. These additional sources proved to be instrumental in defining basic concepts.

Time, the most critical resource in this research, was also the most scarce. The time limitation with respect to expert interviews has already been discussed, but additional time would have allowed for testing the prototype with the expert and actual users.

Computing facilities were not a problem resource with this study. The hardware at the Naval Postgraduate School (development site) was compatible with the sponsor provided software. Computing resources were available throughout the time of development.

The final item on the list of resources is money. This resource, along with time, proved to be another limiting factor in the development of the expert system. Unanticipated budget cuts did not permit the thesis sponsor to fund the anticipated trips for expert interviews, follow-on sessions, and testing. The commitment to perform this research however, was made prior to the occurrence of these funding shortfalls. Consequently, in

an effort to proceed with the research, the author was able to receive enough local funding for the one time interview session described previously. Although subsequent trips would have been most beneficial, this lone trip to conduct the interviews provided enough of the essential elements to work on an initial prototype.

In summary, the "identification" phase described the main players and their respective roles. In this phase, the research was given a focused direction by identifying the problem sources to be addressed. Finally, the resources required to perform the research were also identified. Once all the elements were in place, they provided the requirements for the next stage--conceptualization.

C. CONCEPTUALIZATION STAGE

The conceptualization phase was characterized by a series of interviews with the expert in order to make more explicit the key concepts and relations uncovered from the problem identification in the first phase. These interviews took place over a three day period. The general nature of the interviews will be described here.

1. Expert Interviews

The interviews took place over a three day period at COMNAVAIRPAC headquarters in San Diego, California. The sessions were extremely constructive, but a bit more complicated than anticipated. The knowledge engineer came into the interviews armed with only a superficial knowledge of SERMIS that was gained from the user's manual. Although this knowledge provided a good working foundation, it was insufficient to facilitate a flowing dialogue between the knowledge engineer and the expert. Often times the knowledge engineer would have to stop the expert in the middle of a discussion and ask for further clarification on unfamiliar concepts. The expert would then direct the knowledge engineer to an appropriate source or provide a separate explanation. This significant difference in respective levels of knowledge made the interview process more

tedious than initially expected, but despite these differences, a significant amount of pertinent information was collected.

As discussed previously in the "Identification section," the initial portion of the interview process was spent identifying problem sources. After these areas were designated, discussion turned towards general concepts of the IMRL generation procedure. These generalities eventually proceeded to more specific items until finishing up the sessions with detailed explanations of the SERMIS data elements involved in the IMRL generation process. Throughout the course of these discussions the knowledge engineer was attempting to construct a structured narrative, that is, a "walk through" of the actual steps involved with this process. This process is described in the next section.

Prior to moving into a depiction of the process however, it was interesting to note that when asked what was required to become a SERMIS "expert" in the area of IMRL generation. The expert echoed the perspectives from Chapter II and stated that it first required knowing the facts and then acquiring the experience. He stated that one must become intimately familiar with all the textbook references (facts), and then build a rapport with the activities so an experience base can be built up (heuristics).

2. IMRL Generation "walk through"

During the interviews, particular attention was given to the sequence of events involved in an actual IMRL generation. The expert explained how this process worked prior to SERMIS and how the system operates now, under SERMIS. By comparing the two methods and becoming familiar with them, the knowledge engineer hoped to gain a user's perspective for which steps would be suited for an expert system application. A generic IMRL generation process will now be discussed. Figure 9 summarizes this process.

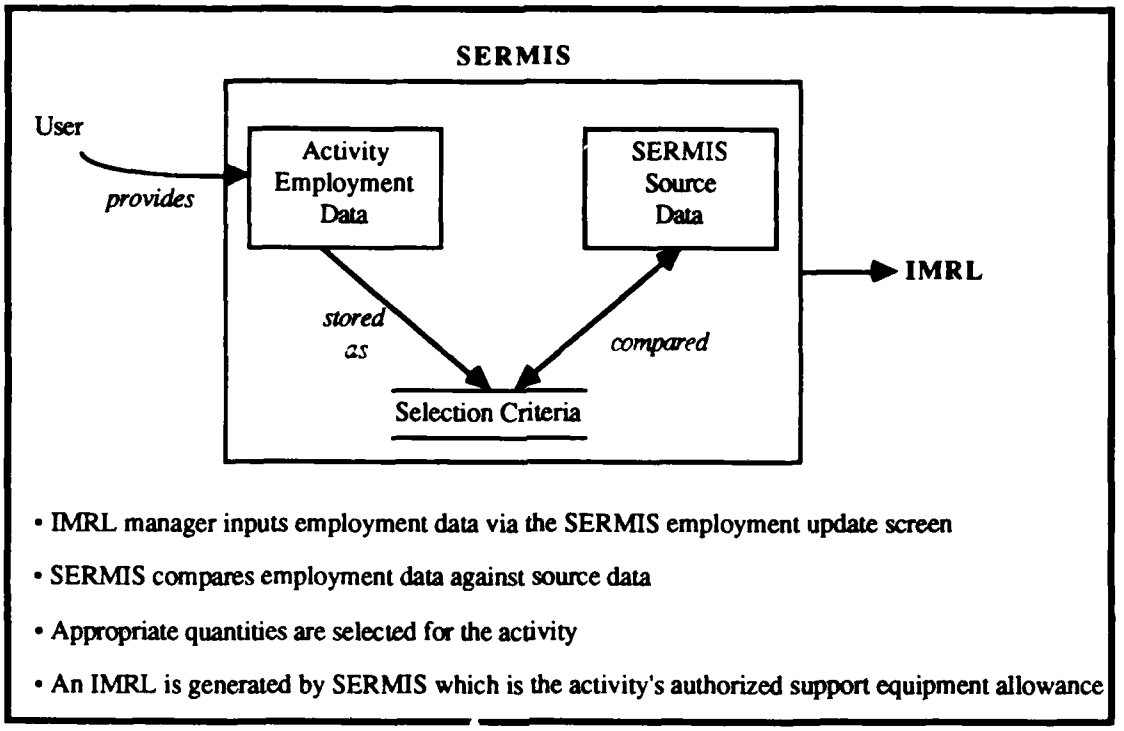

Figure 9. The IMRL Generation Process.

The SECA IMRL manager is responsible for an activity's IMRL preparation. In order to produce this IMRL, SERMIS requires input data both from the IMRL manager and the SECA source data. The source data is transparent to the user. This data contains the "range of allowance" tables. Essentially, there are range of allowance tables for each item of support equipment. To determine what quantity to select from this range of allowances in order for an activity to support a particular aircraft engine for example, SERMIS needs selection criteria. This selection criteria is called an activity's "employment data" and is provided by the IMRL manager. Figure 10 gives some examples of items that comprise employment data. Appendix C provides an explanation of these items as well as their data element names as used by SERMIS.

• Type of aircraft, engine, etc. supported	• Number of aircraft supported
• Activity's level of maintenance	• Selection control code
• Degree of engine repair	• Number of detachments

Figure 10. Examples of Employment Data.

3. Conceptualization Summary

The three day expert interview session was the essence of the conceptualization stage. By discussing the IMRL generation procedure with an expert, the knowledge engineer was able to determine key concepts that should be addressed by the prototype expert system. The concepts decided on in this case were the elements comprising the employment data. If the civilian IMRL manager does not possess enough expertise regarding the SECA's cognizant activities, the potential exists to enter erroneous employment data. SERMIS will still generate an IMRL in a very efficient manner, but it will give inadequate levels of support equipment since the system was provided bad selection criteria. This prototype must be able to provide the civilian IMRL manager at the SECA, the equivalent of a military expert's knowledge regarding an activity's employment data.

D. FORMALIZATION

This stage "involves mapping the key concepts...isolated during conceptualization into more formal representations [Hayes-Roth 84:p.44]." The main intent of this stage then, was to come up with a basic structure that demonstrates how COMNAVAIRPAC's activity employment data can be captured in a knowledge base file. Particularly, the type of knowledge that one acquires after years of experience with the various maintenance activities. This captured knowledge would be made available to the SECA IMRL manager who can then "consult" this knowledge base prior to an actual IMRL generation.

1. Prior Work

The first step in this process was to avoid redundancy of effort. As described in Chapter I, the thesis sponsor has already produced some SERMIS knowledge bases with VP-Expert. The knowledge engineer reviewed the IMRL related prototypes produced by these efforts in order to determine which areas would still require applications. It appeared that a majority of the "textbook" knowledge regarding employment data had already been captured. Figure 11 shows three successive screens from this prior work. The user selections are in bold type, with the dashed lines separating different screens.

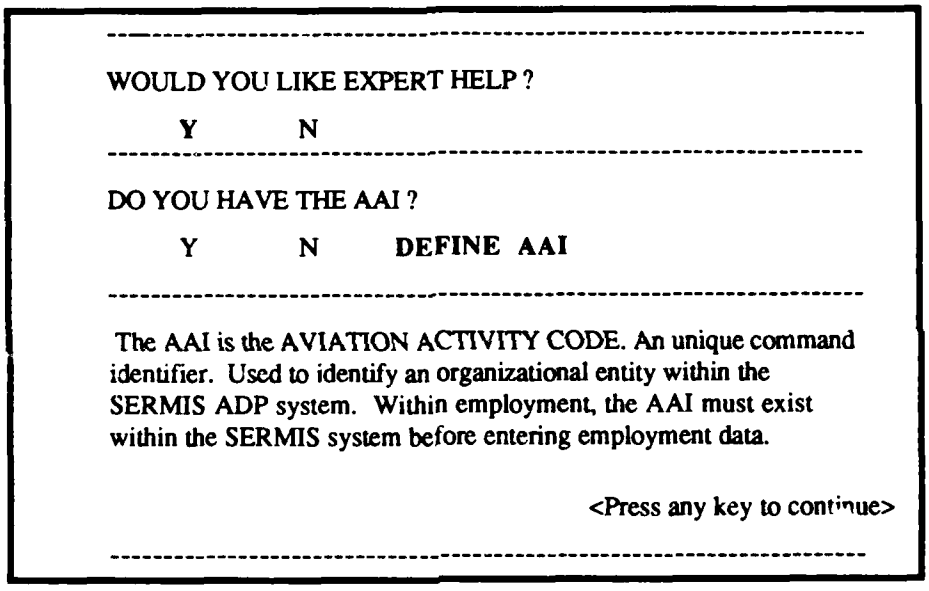

Figure 11. Sample of Prior Work.

This system leads the user through a sequential listing of each element of employment data in the same fashion as depicted in Figure 11. In cases where a valid response can be found in documentation, the user is directed to the applicable publication by the system.

2. Representing Current Information

These initial prototypes are excellent for defining the steps and definitions involved in this process. However, these systems only *refer* the user to a source in order to obtain the required information. After a number of references the user would eventually build experience. The next logical progression in the SERMIS expert system development is then to capture the knowledge from these various outside sources so that this experience is accessible on demand.

In order to accomplish this, the data had to be represented in a manner compatible with "VP-Expert." VP-Expert is a rule-based expert system development tool produced by Paperback Software (a general description of this commercial application along with some of its features are provided in Appendix D). Attempting to encode every data element for each of the AIRPAC activities with a separate rule however would be a formidable task. The VP-Expert architecture provides a more efficient method of encoding the data. Figure 12 gives a schematic representation of a VP-Expert knowledge base.

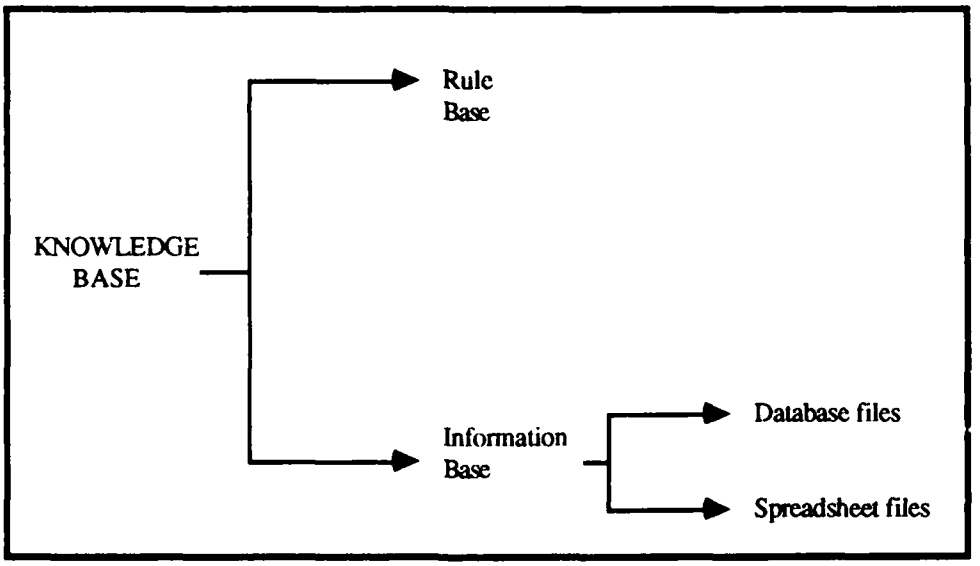

Figure 12. VP-Expert Knowledge Base Components

The two primary components of data storage within a VP-Expert knowledge base are a rule base and an information base. The logic of the rule base has already been described as a series of "IF-THEN" statements. The information base permits large amounts of data to be stored separately from the rule base. This separate data repository can be accomplished through the use of database files and/or spreadsheet files. Collective information on an activity for this research will be kept on a database file. VP-Expert provides a means through which the knowledge contained in the database can interact with the rules kept in the rule base. This feature keeps the rule base from becoming too unwieldy.

This was the strategy taken for data formalization on this project. Activities within COMNAVAIRPAC's SECA would be arranged in database files along with certain types of employment data. Rules would determine what information should be extracted from the database files. Rules would also be used for the incorporation of the less concrete, or heuristic type knowledge regarding the activity.

3. Formalization Summary

The formalization stage was characterized by comparing the concepts to be encoded with the architecture of the VP-Expert program. Methods were then determined on how best to incorporate the different types of knowledge required by the prototype. The two primary methods decided upon were database files and rules.

E. IMPLEMENTATION

This stage involved the actual encoding of the formalized knowledge from the prior stage into the VP-Expert framework. Rules and DBase III Plus files were used during implementation. This section will detail the specific types of information encoded and how it was performed. The proposed implementation of the system within SERMIS will also be addressed. Documentation for this initial prototype is provided in Appendix E.

1. Information Encoded

The amount of information obtained was another casualty of the time/funding limitations of this project. Ideally, the knowledge engineer envisioned sitting down with the expert and examining on an individual basis each of the over 350 COMNAVAIRPAC maintenance activities. The knowledge engineer would be attempting to draw out any unique characteristics for an activity based on the expert's long history of interactions with them. Since there were no additional occasions in which to accomplish this rather ambitious task, the knowledge engineer made note of the isolated examples used by the expert during the course of the interview sessions (VXE-6 was one such example--see Appendix C). To supplement these examples, the knowledge engineer also drew upon personal experiences from the naval aviation community to come up with parallel examples of non-documented activity-specific data that is vital in an IMRL generation. As an example, the knowledge engineer knew from his experience that the F-14A model aircraft is currently receiving completely new engines. The aircraft that receive these new engines also receive a new designation--the "F-14A+." This "plus" designation may appear as a subtle change in terms of semantics, but it has significant impact in terms of support equipment. The F-14A+ has not been delivered to Pacific fleet squadrons as yet, but when it is, IMRL changes will be required. The civilian IMRL managers may not be aware of such transitions. If this information was associated with the activity in the knowledge base, it would not have to be "assumed" information. This type of information was targeted for incorporation into the expert system.

After identifying some of the personal experiences in dealing with maintenance activities, the next priority was to decide which type of activity specific employment data, that is, "textbook" data, could be encoded.

The majority of the data elements that make up the employment data for an activity are situation dependent. Thus it would be a difficult task to cover all valid contingencies with a comprehensive rule set. For instance, an intermediate maintenance activity may support 15 squadrons consisting of three different model aircraft. Employment data such as model number, model quantity and list series selection code can cover a wide range of alternatives. Consequently, in order to demonstrate project feasibility it was decided to use the more static data elements of the employment data. In this case these data elements included the Control Identifier (CI), AMMRL Activity Identifier (AAI) and the Three Degree Code (3 D Code). The Three Degree Code was a good candidate element to include since it was referred to by the expert as a "continuous stickler" when it came to IMRL generation problems.

2. Encoding Process

After the information types were specified, an algorithm was created to identify the sequence of events during program execution. The algorithm is provided here:

- Get activity name
- Provide name along with CI and AAI to the user
- Provide 3 Degree Code of engine repair for the selected activity
- Provide supported/supporting activities associated with the selected activity
- Provide a remarks section concerning the selected activity

To implement this algorithm the first task was to create a database containing the 369 COMNAVAIRPAC activities. This file would be used at the session start to present a master listing of all activity names. The user could then select which activity was desired for an IMRL generation or update. Given the quantity of activities however, all of them could not be presented on one screen and VP-Expert limitations did not provide for a scrolling option. Consequently this master data base information was copied into eight

individual databases, each containing approximately 45 names. The user could then choose a range of activity names from which to select a specific activity. This opening process was considered an improvement since the present method required by SERMIS calls for the user to enter in an AAI number for the activity. Activity names for this menu driven selection method were taken from an April 1988 SECA activity report obtained while at COMNAVAIRPAC. Once the activity name was obtained, the associated AAI and CI would then be presented on screen along with the name. A sample portion of the rule base for this part of the process is presented in Figure 13.

```
ACTIONS
        FIND menu_option
        FIND right_menu
                .
                .
                .
RULE 1
        IF menu_option = A6E_Planning_to_HAMS_13
        THEN
          display the range of names...
    .
    .     right_menu = (menu_option)
    .
    .
ASK menu_option "Select the range for the desired activity:"
CHOICES menu_option: A6E_Planning _to_HAMS_13, HAMS_15_to_HMH_463.....
```

Figure 13. Rule Depiction for Menu Display

In this example from Figure 13, the "Actions" block tells the program to find the goal variable of "menu_option." After searching the rules and coming up empty handed for this value, the system asks the user (via the "ASK" and "CHOICES" statements) to provide a value. In this case it will be assumed that the "A6E_Planning_to_HAMS_13" range of names was the desired selection. Once this value is provided, the expert system is next instructed to find a value for "right_menu." It locates a rule conclusion with this

variable and checks to see if the premise is a valid one. In this case the premise to Rule 1 is now satisfied so the program will execute the pseudocode (shown in italics).

After obtaining the activity name, the next step asked the user if the three degree codes of engine repair were desired. To provide this data, another information base or database file, was created. This file contained the activity name along with each type of engine and the respective degree of repair capability. Once again rules were used to determine when and where to look for this information. A sample rule from this section is provided in Figure 14.

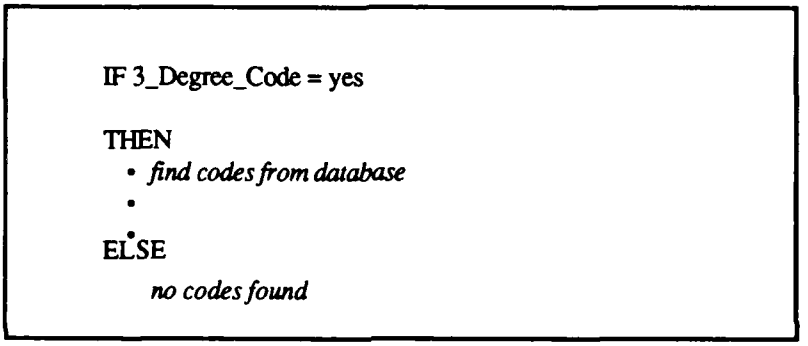

Figure 14. Sample Rule to Find Three Degree Codes

Another essential user provided input for SERMIS is the issue of the equal/unequal CI/AAI. Basically, an activity with an "equal CI/AAI" is a supporting activity and one with an "unequal CI/AAI" is a supported activity. The SERMIS manual provides this example:

> NAS North Island AIMD is CI 00246 and AAI 00246. This will be referred to as the equal CI/AAI or the supporting activity. The squadron is the unequal CI/AAI. For example, HSL-41 FRAMP North Island is CI 00246 and AAI 55139. This will be referred to as the unequal CI/AAI or the supported activity [SERMIS 86:p.C-15].

The *supported* activity requires the use of certain support equipment maintained by the *supporting* activity. The use of an unequal CI/AAI along with the type IMRL selection code, will ensure appropriate items will be posted to the correct activity. Understandably, this concept has great potential for creating confusion. This prototype consults a master database file after an activity selection in order to tell the user whether the selected activity is a supported or a supporting one along with the associated supported/supporting activity name.

As a final step in the gathering of correct employment data, the expert system will consult a separate knowledge base file in order to find any remarks concerning the activity. This separate knowledge base consists entirely of rules that contain information on an activity. The logic behind a separate knowledge base for remarks was to keep the system as a modular design thus making system maintenance easier. If an activity's remarks require updating, it can be done without having to enter the main program.

3. Prototype Use

The prototype as an entity, is designed to be used as a front-end application prior to an IMRL generation. It is believed that the IMRL manager would run the program on a personal computer adjacent to the SERMIS terminal. With this arrangement, the user could gather all the captured information on an activity prior to SERMIS data entry. Even prior to committing the data from the SERMIS screen, the expert system could still provide a ready reference source in the event of last minute questions.

4. Implementation Stage Summary

In summary, implementation was achieved through the use of database files (information bases), rule bases, and a separate knowledge base file. The information bases contained "hard data" on the activities; the main rule base contained the rules that controlled

the access and manipulation of the the information bases; and the separate rule base held the remarks, or the more dynamic data on the activity.

F. TESTING

The final stage in the iterative construction of a prototype is testing. The strategy for this project used four of testing. These testing stages, as described by Pressman, are: unit, integration, validation and system [Pressman 87]. Each of these stages will be addressed in this section. Table 7 provides a summary of these testing levels and the specific areas tested in each level with respect to this project. The shaded area indicates the levels that have not been tested to date.

TABLE 7. Testing Strategy Summary

TEST TYPE	AREA TESTED	SERMIS EXAMPLE
Unit Testing	Code	Rules
Integration Testing	Module interfaces	Interfaces between knowledge base and databases
Validation Testing	User requirements	Will it provide accurate/useful employment data
System Testing	Incorporation into the overall system	Fit into the SERMIS program

1. Unit Testing

With unit testing, emphasis is on the individual modules. Data structures and integrity are tested by executing the statements within the module. In this project, unit tests were done to validate the rules contained in the knowledge base files. Test cases were developed to verify:

- Rule premises
- Rule logical operators
- Loops

Rule premises (the "IF" portions of a rules) were tested by using known activity employment data and comparing it to the employment data provided by the expert system. These cases were used to check if the inference engine was in fact "firing" a rule when the premise was valid. If system did not provide the same information as known beforehand, the knowledge engineer would manually "backchain" through the code to determine which rule or rules were at fault. Conversely, a check was also done to determine if any rule conclusions were being implemented when the rule premise was false, such as, listing three degree codes for an activity when in fact none applied.

There were some occasions where multiple conditions had to be satisfied either for a rule to be considered valid or for a database record to be selected. These criteria were based on conditions linked by the "AND/OR" logical operators. An example of such a rule premise would be: IF activity = USS Enterprise OR USS Nimitz. Test cases were run that verified each one of the valid options would satisfy the rule premise.

In instances where the expert system had to retrieve successive records from a database file, a loop condition was set up. The loops were executed with test cases to ensure: all valid responses were being listed; a maximum of one screen of information was being displayed at a time; and that there were not any infinite loop conditions.

2. Integration Testing

Pressman defines integration testing as "a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing [Pressman 87:p.507]." Within this project, interfaces had to be tested between the two knowledge base files and the various database files. To accomplish this level of testing, a "top-down" approach was used. Top-down is a methodology that starts with the main control module (in this case, the main knowledge base file) and tests the interfaces by substituting "stubs" for the subordinate modules. In this research, stubs were

used to replace database files and the subordinate knowledge base file. Representative test data was then passed across these interfaces to verify the data integrity. Once this process was validated, the respective database file or knowledge base file was constructed replacing the test stub.

3. Validation Testing

Validation testing ensures that the software has met the user requirements. Boehm rather succinctly and accurately captures the intent of this phase with the question "Are we building the right product?" [Boehm 81]. Testing in this phase would have required additional sessions with the expert in order to determine if the information being provided by the expert system was indeed correct. The project limitations precluded testing at this level however. Once the prototype is turned over to the sponsor, validation testing can be performed at his discretion.

4. System Testing

System testing occurs after the expert system software has been completely validated. Once this has been completed, it can now be incorporated into the SERMIS environment. System testing checks for the problems created by this integrating of systems. System testing was also not accomplished within this project for the same reasons that addressed in validation testing.

G. CHAPTER SUMMARY

The design and construction of this prototype followed the five stages of knowledge acquisition as described by Hayes-Roth. These stages were: Identification, Conceptualization, Formalization, Implementation, and Testing. This process began with a sponsor interview and a SERMIS user's manual. Following exposure to the manual the knowledge engineer proceeded into the enlightening, and perhaps somewhat inelegant expert interviews. This one shot opportunity at the expert did however, identify problem

origins to be attacked, and enough expertise and documented source material for the knowledge engineer to start work on a prototype. Emphasis was placed on SERMIS employment data and the VP-Expert knowledge base structure. A large portion of the data was incorporated into DBase III Plus files referred to as "information bases," with the remainder of the data being encoded into rules. Only the first two levels of testing--unit and integration--were performed for this project. Unanticipated funding cutbacks and schedule limitations prevented testing of the validation and system levels.

IV. FINDINGS, CONCLUSIONS, AND RECOMMENDATIONS FOR FUTURE STUDY

This chapter will review the project scope, the problem addressed by the prototype, and the methodology involved. In the results discussion, the original research questions will be answered as well as addressing the overall conclusions as seen from the perspective of a novice knowledge engineer. Finally, requirements for follow-on development will be discussed.

A. RESEARCH SUMMARY

This research was initiated in response to a Naval Aviation Maintenance Office request for the development of an expert system application. This application would be used to gain a more effective employment of SERMIS. The specific area of IMRL generation was chosen for this application because there is either: no one with enough experience to fill the vacant IMRL manager positions; or the individual presently filling the position does not possess enough background experience to accurately input all activity employment data.

The thesis sponsor provided the VP-Expert development tool and scheduled time for a brief interview period with a recognized expert. The author acted in the capacity of knowledge engineer and used a prototyping methodology that consisted of the following phases:

- Identification
- Conceptualization
- Formalization
- Implementation
- Testing

The prototyping or iterative approach was considered appropriate due to the heavy interactions with the human expert and the functionality of an expert system could be demonstrated without a prolonged effort [Pressman 87].

B. FINDINGS

One of the main questions to be answered by this effort was the identification of the unique knowledge requirements necessary to produce a complete and accurate support equipment allowance. The expert interviews revealed that a greater personal awareness of an activity's employment data is the distinguishing factor. The difficult aspect in acquiring this knowledge was not due to any conceptual complexities in learning this information, but rather, just the sheer quantity of required information that exists regarding all the activities in the COMNAVAIRPAC SECA. So there does exist a quantifiable body of knowledge. Once this knowledge, collectively known as employment data, becomes second nature to the IMRL manager, it has the capability of making that person an "expert" in the area of IMRL generation. This body of knowledge is comprised mainly of facts and "textbook" data. Examples would include: Three Degree Code of engine repair and whether the activity is a *supported* or *supporting* activity. Some of the knowledge however, is gained through benefit of continuous interactions with the subordinate activities within the SECA. This experience provides insights into an activity's operating environment, such as knowing the activity now belongs to a new airwing, or that a new engine is coming out for its model aircraft.

The previous finding indicated that the expert knowledge is identifiable. The next question then becomes: can a simple expert system be constructed that captures this knowledge? This research showed the answer to be yes. VP-Expert's architecture lent itself to this task very nicely. Since a majority of the knowledge was factual, it could be

stored in database files that in turn could be accessed by the main knowledge base program. Information such as equal or unequal CI/AAI, Three Degree Codes, and activity names were all incorporated into database files. The "heuristic," or mostly private information on activities, was encoded into rule conclusions to be displayed as on-screen text. If the rule premise was proved true, such as "if activity = F14," then the "expert advisory" regarding an F14 activity would be displayed for the user.

Another observation from this project regarded the appropriateness of VP-Expert as a development tool. Was it effective in knowledge acquisition? Overall, it did reduce the "bottleneck" commonly associated with the knowledge acquisition process. To realize the true effectiveness of such a tool however, one must still have valid concepts to encode. A development tool such as VP-Expert, is not a panacea for the problems associated with the knowledge acquisition process. There still is no substitute for human interaction when it comes to eliciting the fundamental concepts from another human. Once these concepts were known, VP-Expert proved to be an effective and valuable development tool. The transitions from concepts to formalization to implementation were made in a much smoother and more time efficient manner using this program.

Another finding this research set out to address was the integration of an expert system into an existing MIS. This first cut prototype obviously never reached a mature enough stage to warrant an immediate incorporation into SERMIS. Chapter II addressed two strategies however to consider when the time is appropriate for incorporation. The first option is that the expert system can be integrated into existing SERMIS components. The second choice is to keep the expert system as a separate component. While development continues, this prototype should certainly be kept as a separate component. This approach would have no impact on SERMIS during the frequent expert system refinements being performed during the course of development. Even when the expert system achieves a

relatively stable existence following development, most of its advantages will still be gained through use as a separate, or front-end component. The conversion of the expert system's MS DOS database files and knowledge bases to a mainframe environment will simply be too complex a process to support when changes are required.

C. REQUIREMENTS FOR FOLLOW-ON DEVELOPMENT

The first priority in a follow-on effort is testing. The validation stage of testing the initial work must be accomplished prior to moving on to future prototype iterations. Additional time with the expert should be arranged in order to conduct this validation testing. This testing should not be limited to just the expert either, inputs should also be sought from actual SERMIS users. The expert's scrutiny will help validate the information and concepts being captured, while the user's inputs will indicate if this knowledge is in fact beneficial in the execution of the IMRL generation process.

More interactions between expert, knowledge engineer, and user will permit expanding on the original concepts. This entails more interview sessions with the expert . In the absence of follow-on interviews for this project, the knowledge engineer drew upon personal experiences to supplement the captured information. Eventually, as more difficult concepts emerge, more time must be scheduled with the expert. Additional VP-Expert features, such as confidence factors can then be employed to help structure the more complicated concepts.

Expert system feasibility has been proven by this effort and the structure is now in place. Follow-on efforts are required to supplement the initial activity information provided from documentation and the limited sessions with the expert.

D. GENERAL CONCLUSIONS

The main intent of this thesis was to provide the author a more comprehensive background in expert systems. Prior to this research the author's exposure to expert systems had been purely academic. This project was a vehicle to bridge that space between theory and experience. It moved the learning experience from the classroom to a real world application through a knowledge acquisition process for a prototype expert system development. The additional goal during the course of this process was to provide the thesis sponsor with a prototype SERMIS application.

The position of knowledge engineer is still being defined in this, the early stages of expert system applications. It is a demanding position with relatively few ground rules. This research reaffirmed the complexities inherent in the position as well as the wide range of skills required for the task of knowledge engineer. The knowledge engineer is the common denominator throughout all stages of expert system development. Each stage focuses on a different ability. At the outset, the knowledge engineer must be a student, by learning the basic concepts that are involved in the proposed expert system application area. The more fundamental knowledge acquired, the less the knowledge disparity between the expert and the developer when it comes time for the knowledge extraction sessions. Once the expert interviews commence, time becomes a critical resource. The knowledge engineer must possess competent communication skills in order to maximize the available time. If the knowledge engineer is attempting to draw out over 30 years of accumulated knowledge from the expert, every minute becomes valuable. After identifying the essential concepts from the expert, the knowledge engineer must exercise technical skills to encode these concepts into data structures.

In summary, the first attempt at knowledge engineering is an eye-opening experience. It provides a genuine appreciation for the depth of the assignment and reinforces the need to keep the application confined to a narrow knowledge domain.

It was also concluded that the use of an expert system development tool will reduce some of the workload created by the required steps described above. Development tools provide capabilities to help structure concepts, such as databases that can be used as information bases. It is certain that these tools will continue to develop more potential with time and as a result, the knowledge engineer's life will be made a little easier. This project would have been infeasible for a first time developer without the aid of a development tool.

Finally, the value of the prototyping methodology was also proven by this research. Hayes-Roth et al. describe one of the most important advantages of prototyping:

> The development of the prototype system is an extremely important step in the expert system continuation process. Some code may be salvaged from this throw-away program for later versions, but the most important part of the exercise is testing the adequacy of the formalization and of the basic underlying ideas [Hayes-Roth 83:p.146].

So it was with this research. It was never intended to produce perfect coding in order to provide direct implementation of an expert system to the sponsor. Rather the concepts considered crucial to capturing the required information were demonstrated by the prototype. It is up to follow-on efforts to build on these established concepts and refine the code.

E. APPLICATIONS FOR OTHER AREAS

Even though the ultimate question of whether this research will eventually lead to a fully developed system that improves the IMRL generation procedure is still to be

answered, a prototype proved the basic potential of expert systems. The advantages of this technology can be extrapolated to other areas.

One such area, particularly appropriate to the military environment, is where the expertise is required, but the expert is gone. Such an example, as described in Chapter I, is the case of taking over a billet after the predecessor has departed. By having the corporate knowledge stored in an expert system, the job transition can be still be accomplished effectively. As expert system development tools become more powerful and more user friendly, the potential exists for the "expert" to maintain his own expert system.

In today's environment, the jobs requiring the use of an automated technology stretch all the way down to the operational level. In the military, the use of such technology, whether it is word processing or weapon systems, is often in the hands of younger and less educated personnel. Expert systems can be employed to train this technologically inexperienced work force. Expert systems can be also be used (as it is intended for SERMIS) as front-end applications that make the operation of a complex system more "user friendly."

As the number of automated systems increase in an environment requiring a rapid response, the time available for decision making decreases. The time previously available to consult resident experts before making a decision may no longer be available. Expert systems can rapidly provide the information when the experts are not immediately available.

F. OVERALL SUMMARY

Expert systems reflect the merging trend of the psychological and computer science fields of study. These systems continually strive to capture the human cognitive process in computer code. As of late, expert systems have been moving rapidly from the research

environment into practical applications. A prototype expert system application was developed as part of this research. It captures the knowledge necessary to generate accurate support equipment allowances under the SERMIS program.

This research identified some of the knowledge, both factual and heuristic, considered integral in accomplishing this task and demonstrated the feasibility of incorporating this body of knowledge into an expert system. The knowledge engineer was the link between this body of knowledge and the expert system. It was also shown that the position of knowledge engineer is a demanding one that requires a wide array of skills. The process of encoding the knowledge was greatly facilitated by the use of VP-Expert. This development tool minimized the overall time of prototype development.

Ultimately, the advantages of expert systems should continue to become more and more visible. As expert system development tools become more powerful, the technology will be easier and more economical to incorporate into a wider array of potential areas.

APPENDIX A

SERMIS OVERVIEW

A. HISTORY

The Support Equipment Management Information System (SERMIS) project has evolved in response to the need for automated support within the Aircraft Maintenance Material Readiness List (AMMRL) Program. The AMMRL program is a Naval Air Systems Command Headquarters managed program for aviation support equipment established in 1960. The goal of this program is to "ensure the availability of support equipment to meet flight and personnel safety requirements; operational readiness, and mission effectiveness goals" [NAVAIRINST 87]. Prior to SERMIS, manual generation of allowance lists were considered inadequate because of different criteria being used and an inability to recognize particular needs of different activities. In 1963 allowance lists were generated by batch operations. The purpose of SERMIS was to move this data generation process to an interactive database management system with real-time capabilities. SERMIS implementation began in 1979.

B. SYSTEM DESCRIPTION

SERMIS has now become the repository of master support equipment data. SERMIS operates from a central data base in New Orleans that recognizes approximately 27,000 items, 600 airframe configurations, 70 power plant configurations and almost 1250 avionics, missiles and armament systems [SERMIS 86]. This program is run off of a Sperry Univac 1100.

SERMIS is composed of six major subsystems:

- INVENTORY-- maintains total reportable assets for each IMRL activity
- SOURCE DATA--maintains the catalog and technical data provided by the Automated Support Equipment Recommendation Data (AUTOSERD) system. AUTOSERD is the sole update medium for SERMIS.
- ALLOWANCE--provides the Support Equipment Controlling Authorities (SECAs) with the n-line capability of creating and maintaining employment data. This employment data combined with established algorithms, compute Support Equipment (SE) allowances.
- SE ASSET READINESS REPORTING--updates and maintains activity descriptive information records for use by other SERMIS subsystems.
- REWORK--tracks all pertinent information required to manage SE end items going through depot level rework.
- SECA TECHNICAL DATA--assigns and monitors special management codes to items not listed in the source data.

C. USERS

The six Support Equipment Controlling Authorities (SECAs),are the primary users of SERMIS. They are:

- Commander Naval Air Forces U.S. Atlantic Fleet (COMNAVAIRLANT)
- Commander Naval Air Forces Pacific Fleet (COMNAVAIRPAC)
- Chief of Naval Air Training (CNATRA)
- Commander Naval Air Reserve Force (COMNAVAIRESFOR)
- Commander Naval Air Systems Command (COMNAVAIRSYSCOM)
- Naval Air Maintenance Training Group (NAMTRAGRU)

These SECAs are major aviation commands that provide administrative control for the allowance and inventory of support equipment end items. These organizations use the data from the central repository in order to provide Individual Material Readiness Lists (IMRLs) and the monthly supplements to Navy and Marine Corps support equipment users.

APPENDIX B

SPONSOR INTERVIEW

This was a one day meeting that took place on May 2, 1988 between the author, and the thesis sponsor of the Naval Aviation Maintenance Office, Patuxent River Naval Air Station, Maryland. The sponsor had solicited for thesis students to do expert systems work for the SERMIS project.

This meeting provided the opportunity for a face-to-face discussion regarding the sponosor's desires for SERMIS and the author's requirements for a thesis. It also allowed the author to pick up a copy of the SERMIS user's manual, and the "VP Expert" software package and documentation.

Much of the discussion centered on SERMIS and its particulars (the author has had first hand experience in naval aviation activities, but no experience with SERMIS). The sponsor summarized the complexity and importance of SERMIS by stating this system controls the $6 - 9 billion inventory of Support Equipment. SERMIS is composed of six subsystems (Appendix A). The sponsor desired the expert system work to be directed towards the Allowance subsystem.

An actual case was related where SERMIS misuse in the Allowance subsystem had strong ramifications. An aircraft carrier was required to make an unscheduled deployment in response to a real world contingency. The airwing, which comprises approximately 95 aircraft, was newly created and consequently required a new IMRL to be generated in order to provide the necessary support equipment to the airwing while deployed on board the carrier. This process was given priority on SERMIS, and a weekend of computing time was spent generating this 3000 plus page document. The end result provided incomplete levels of support equipment, and the entire process had to be performed again.

This real world example led into the discussion of the basic problem: erroneous inputs are creating tremendous deficiencies in support equipment allowances. Now while some of these cases can be solved by repetition, others simply cannot afford to be rerun without mission impact.

The sponsor attributes these end product deficiencies to a lack of expertise on the part of those inputting the parameters. The main users of SERMIS are called Support Equipment Controlling Authorities (SECA). There are six of these SECA activities (see Appendix A for a more detailed description of the SECA's) and they are listed below:

- Commander Naval Air Forces U.S. Atlantic Fleet (COMNAVAIRLANT)
- Commander Naval Air Forces U.S. Pacific Fleet (COMNAVAIRPAC)
- Chief of Naval Air Training (CNATRA)
- Commander Naval Air Reserve Force (COMNAVAIRESFOR)
- Commander Naval Air Systems Command (COMNAVAIRSYSCOM)
- Naval Air Maintenance Training Group (NAMTRAGRU)

Each of the above listed SECA's has a position that carries out the *IMRL* generation and updating for each activity under its control. Some SECA's can have as many as 700 activities under its cognizance. Since automating the support equipment system under SERMIS, these positions have been staffed by civilians. Initially, there were few civilians with enough prior military expertise under the manual system (pre-SERMIS) that would qualify them for this position. There were sufficient numbers to staff each SECA originally, but this initial cadre has gradually retired with no replacements. The sponsor states that there is currently a real scarcity of expertise when it comes to a thorough understanding of the mechanics of support equipment allowances, that is, an understanding that could really only be gained through years of generating IMRL's by hand. The sponsor went on to state that out of the six SECA's, two of them have to borrow the "expert" now when it comes to involved operations.

All of the background discussion was summarized by stating that the knowledge from one of the few remaining experts needs to be captured. This expert should be one who has had direct experience with the entire support equipment program evolution. This knowledge could then be incorporated into a prototype expert system, and used as a front end program to the allowance subsystem.

APPENDIX C

EXPERT INTERVIEWS

A. OVERVIEW

The time dedicated for the travel to San Diego to conduct the expert interviews was only three days. This was the only time available due to academic schedules and funding constraints.

The first session was spent clarifying and gathering SERMIS specific information that was previously unavailable to the researcher. The second session dealt with the actual steps involved in the generation of support equipment allowances. The third and final session discussed the particular data elements that are the user provided inputs to SERMIS for the allowance lists.

B. BACKGROUND OF EXPERT

The designated expert is currently employed by Science Applications International Corporation (SAIC). SAIC is a civilian consultant firm that is contracted to perform work for the Naval Aviation Logistics Center (NAVAVNLOGCEN). The expert's current position is an ADMRL/SERMIS Senior Systems Analyst, which entails providing services for NAVAVNLOGCEN's support equipment management products.

This gentleman qualifies as a SERMIS expert by virtue of over 40 years experience in the aviation support equipment environment. Prior to his position with SAIC, he retired from the Navy as a Senior Chief Aviation Machinist with 20 years of experience in aircraft maintenance and maintenance administration. He then worked in the Federal Civil Service for 17 years at the Naval Air Systems Command Representative Pacific where he administered the Aviation Maintenance Material Readiness List (AMMRL) program.

In summary, he has been described as the only man that has personally experienced all aspects of the AMMRL--from manual methods to SERMIS.

C. FIRST SESSION

The first session began with general introductions, description of backgrounds, and the intent of these interviews. Much of the subsequent discussion centered on clarification and further explanation of specific aspects of SERMIS. During this initial session, the original problem definition of inaccurate or incomplete Individual Material Readiness Lists (IMRL's), that is, support equipment allowances, was made more explicit. The problem was still attributed to two sources: errors in the central database, and errors from user inputs. The errors in the central database are not a part of this study. The interviews did provide the following underlying causes for user errors however:

- Availability: few people are required to have the knowledge to fill the positions. Consequently, when someone does vacate the position, there are no replacements.
- Responsibility: since the IMRL generation process is now performed by civilians, there is no direct appreciation of the consequences of their actions on operational units, in other words, the IMRL managers at the SECA level do not live with the results.
- Job appeal: this particular SERMIS responsibility was described as "not a glamorous job," since it more often than not invokes criticism instead of compliments.

After elaborating on the problem sources, the interview turned towards a discussion of the prerequisite knowledge involved in the generation of support equipment allowances. The resident expert stated that the primary building block is OPNAVINST 4790.2, or the "maintenance bible." This publication details maintenance functions and assignments of responsibilities. Once an operating knowledge is acquired with this formal publication, one must establish a rapport with the different types of activities within a Support Equipment Controlling Authority's (SECA) jurisdiction. For example, VXE-6 is a C-130 squadron that is tasked to fly research missions. There are times when these missions are flown

from the Antarctic. When the squadron is in the United States, it is considered an organizational level maintenance activity. When the squadron operates out of the Antarctic, they are considered an intermediate level activity since they must support themselves in a remote location. Support equipment allowances will vary significantly depending on this particular squadron's base of operations. By building this rapport with the maintenance activities, one becomes aware of these various subtleties that potentially have a significant impact on allowances.

This was just one example of the type of cases the operator at SECA must be aware of prior to sitting down in front of the SERMIS terminal and typing in the data elements required for an IMRL for a particular activity. The bottom line is that even though the allowance process is now automated, one still has to do all the necessary "homework" on the respective activity to ensure valid input parameters.

D. SECOND SESSION

After the review of problems, concepts and hard copy sources from the initial session, the second session was spent reviewing actual allowance reports and describing the general mechanics of an IMRL generation under SERMIS.

SERMIS draws on two inputs when generating an IMRL for a specific activity. The first input comes from the source data and is not subject to any user manipulation. Basically, the source data contains all the prerequisite information such as range of allowance tables. The second input is provided by the user via the "activity employment update" screen in the Allowance subsystem. Figure C-1 shows a sample of this screen [SERMIS 86:C-33]. The underlined items (see "Third Session" section for explanation of these items) concern information on an activity's configuration such as number and types of entities supported [NAVAIRINST 87]. Once the user is satisfied with these inputs, they

can be committed and SERMIS will use this information to update/create the IMRL for that activity.

ACTIVITY EMPLOYMENT UPDATE

AAI KA-6D USS CORAL SEA CV-43 CI 03343 USS CORAL SEA CV-43

ACT CODE	SUB IND	TYPE IMRL SEL CODE	SLM NUMBER	SLM QTY	M/L CODE	LIST SERIES SEL CODE	3 D CODE	NUMBER OF DET
V	N	C	KA-6D	005	0	00	N	- -

EXCEPTIONS

Figure C-1. Sample of an Activity Employment Update Screen

The entire IMRL generation process can be summarized in these steps:

- The user gathers all the pertinent data regarding the activity, such as level of maintenance performed and types of aircraft supported.
- The user then calls up the "Activity Employment Update Screen" and enters the required information. This screen is considered a "work area" until the user instructs SERMIS to incorporate the data.
- Once the information is committed, SERMIS performs an editing and validation check on the data, and if correct proceeds to the source data to apply this information to the algorithms and allowance tables.
- An IMRL is then established for this activity which can be tailored by the SECA through another SERMIS application.

E. THIRD SESSION

The final interview session consisted of more research on topics covered in the previous sessions, and explanations from the expert on the data elements that the user

inputs to SERMIS via the screen depicted in Figure C-1. The data element names and meanings from that screen are summarized below:

- CI--Control Identifier. A code assigned to each activity in SERMIS.
- AAI--AMMRL Activity Identifier. A number used to identify an activity which submits inputs or receives outputs from SERMIS.
- ACT CODE--Activity code. Used to specify a land or vessel activity (L or V).
- SUB IND--Subcustody Indicator. Used when activity is supported by multiple supporting activities. This was a category designed for a large number squadrons in close proximity on the east coast. The COMNAVAIRPAC SECA always uses "N"(No).
- TYPE IMRL SEL CODE--Type IMRL selection code. It serves as a means of controlling the selection of support equipment from the source data. In this case "A" selects both intermediate and organizational level items. B,C and D are used less frequently since they cover unique cases.
- SLM NUMBER-- System/List/Model number can be either a: system number; list code; or a model designation (as shown in Figure C-1) to be supported by an activity.
- SLM QTY--System/List/Model quantity. A numeric value greater than zero to reflect the quantity of systems supported.
- M/L CODE--Maintenance Level Code. This code is used to identify the level of maintenance performed by an activity (D=Depot, I=Intermediate, O= organizational, T = transient).
- LIST SERIES SEL CODE--List series selection code. This is used to control the selection of SERMIS source data. For example, "00" will select all appropriate items (list codes) needed to support the model or system at the activity, while "TA" will cause a selection of only the airframe items required.
- 3 D CODE--Three Degree Code. This is a code that denotes what degree (first, second, or third) of engine maintenance is performed at that activity. Different criteria will be used by the source data depending on the code entered.
- NUMBER OF DET--Number of detachments. A number that specifies how many detachments are within this parent activity. In this example a blank indicates that this category is not applicable.

The question was asked if there were any historic "problem" data elements from the above list. The expert felt that the Three Degree Code of engine repair was probably the worst offender that he could recall, but in general there was really no specific set of data elements that consistently caused errors in the allowance process. The expert re-emphasized

the need for the user to become really involved with the process, that is, to understand the workings and subtleties of each activity and to do the appropriate amount of research on the activity prior to sitting in front of the terminal. Other areas he believed that would benefit from an expert system would be activity related specifics as: types of aircraft supported, level of maintenance performed, airwing composition and so on.

The remainder of the time spent at COMNAVAIRPAC consisted of gathering additional publications and references that would serve as source material for some of these activity specific items.

APPENDIX D

VP-EXPERT

A. DESCRIPTION

VP-Expert is a microcomputer-based expert system development tool. This software runs on an IBM PC, PC-XT, PC-AT or compatible system. It requires 256K or more of RAM and is produced by Paperback Software International of Berkeley, California.

B. FEATURES

Listed below are some of the main features of VP-Expert :

- Inference Engine. This inference engine makes use of both the backward and forward chaining methods (see Chapter II).
- Knowledge Base. The knowledge representation method used by VP-Expert is rule-based, that is, a series of "IF-THEN" statements. The knowledge base also contains statements that help structure the problem solving process. For example, statements are used to present menus, ask questions of the user, or to identify a goal variable.
- Explanation facility. VP-Expert provides options that allow a user to "view" the problem solving session. A "Rules" window shows the knowledge base rules that are being processed by the inference engine. The "Results" window depicts the conclusions (intermediate and final), or values assigned to the variables along with the confidence factors.
- Confidence factors. Confidence factors (0-100) can be applied to the rules in the knowledge base and to the user provided inputs.
- Induce function. This option allows creation of a simple knowledge base from an induction table that is created in the text editor or from a compatible data base file.
- Text Editor. This editor can be used for the creation and editing of the knowledge base. It enhances error correction ability by allowing the user to immediately enter the knowledge base after conducting a run in order to make the appropriate corrections.

C. KNOWLEDGE BASE

In order to perform a consultation with VP-Expert, one requires a knowledge base file. This file contains the instructions and rules pertinent to a specific consultation. Each knowledge base file contains three basic elements:

- The ACTIONS block
- Rules
- Statements

The "ACTIONS" block of the knowledge base can be thought of as a small program. It contains an ordered list of tasks for the expert system to carry out. In the example used in Section D, there is only one task.

Rules are the series of "IF-THEN" statements that contain the encoded knowledge. Rules may occur in any order within the knowledge base.

Statements are additional code in knowledge base that provide additional information pertinent to the consultation that are not in the rules. For example, the "ASK" statement allows the system to ask questions of the user, and a "CHOICES" statement will then present a menu of choices fro. which to select.

D. VP-EXPERT EXAMPLE

This section will present a simple example taken from the VP-Expert user's manual in order to illustrate some of the above concepts [VP-Expert 87: 2.6-2.10].

Figure D-1 shows a simple knowledge base file that consists of three statements and four rules that would be used to find an appropriate cheese to serve with a particular course.

```
ACTIONS
                FIND The_Cheese;

RULE 1
IF              Complement = crackers_and_bread
THEN            The_Cheese = Brie CNF 80;

RULE 2
IF              Complement = bread_and_fruit
THEN            The_Cheese = Cambert CNF 100;

RULE 3
IF              Course = Appetizer
THEN            Complement = crackers_and_bread CNF 100;

RULE 4
IF              Course = Dessert
THEN            Complement = bread_and_fruit CNF 100;

ASK Course: "Is the course Appetizer or Dessert?";

CHOICES Course: Appetizer, Dessert;
```

Figure D-1. A VP-Expert Knowledge Base File Example.

Here, the system was instructed by the "ACTIONS" statement to find a cheese to recommend for a particular course. The inference engine looks for the first rule with the goal variable, "The_Cheese" in its conclusion. In this case it is Rule 1. Once the rule is found, the inference engine will attempt to test the rule, but in this case the value of "Complement," in the premise, is not known, so the inference engine cannot test it for the time being. In order to do so, it searches next for a value for "Complement" by looking for the first rule with "Complement" in the premise. This brings the inference engine to Rule 3, and once again it attempts to test this rule. Once again, the rule cannot be tested since the value for the premise variable, "Course" is unknown. In order to test this rule then, the inference engine will attempt to find a rule with "Course" in its conclusion. In this example however, there are no rules with "Course" as a conclusion. If the inference

engine cannot locate a rule that assigns a value to "Course," it will look for an "ASK" statement. In this sample problem, it finds one so VP-Expert displays the ASK statement to the user. The "CHOICES Course" statement will cause the choices of Appetizer and Dessert to be displayed with the ASK statement. The user can now enter a choice, and the inference engine can work back through and test the rules.

The "CNF" designation from the above example illustrates the use of confidence factors. The confidence factor of 80 from the rule 1 conclusion means that this conclusion is drawn with 80% confidence. Confidence factors should not be confused with statistical probability since the confidence factors are subjective assessments. If no CNF value is entered, the default value of 100 will be assigned. In this example, the values of 100 in the last three rules were added for puposes of illustration.

Since the system started with a goal and worked backwards to determine which rules to test, the system made use of backward chaining. Figure D-2 shows a sample screen from this consultation. The upper portion is the consultation window, the lower left section is the "rules" window, and the lower right is the "results" window.

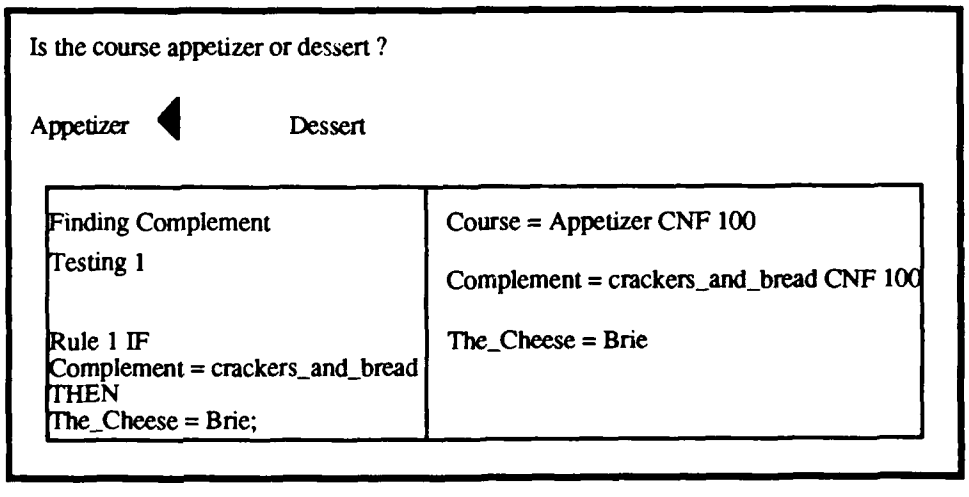

Figure D-2. VP-Expert Sample Screen

E. SUMMARY

VP-Expert is a rule-based expert system development tool. The user employs the text editor to develop the knowledge base by using "IF-THEN" rules as well as VP-Expert specific statements that perform such functions as: identifying goal variables, presenting menu choices, assigning confidence factors and so on. The built-in inference engine will then go through and test the rules (normally using backward chaining) to arrive at a conclusion.

APPENDIX E

DOCUMENTATION

A. OVERVIEW

This appendix provides general instructions for the use of the COMNAVAIRPAC SERMIS expert system application. It is assumed the user will be familiar with the general procedures of VP-Expert. A description of the different files (database and knowledge base) used in the construction of the first version prototype is also provided. This appendix is also intended to be used as a reference for any follow-on work with respect to this program.

B. START UP INSTRUCTIONS

All the files necessary to run this prototype program are on the disk that is labelled "AIRPAC.KBS". It is designed for use on a computer system with two disk drives. The VP-Expert system disk should be inserted into drive "A" and the AIRPAC.kbs disk should be inserted into drive "B". Start the VP-Expert session as normal, choose the "Consult" option, and then type in "B:Airpac" when prompted to enter a knowledge base file name. The system will then begin the consultation.

C. CONSULTATION DESCRIPTION

The main intent of this prototype was to demonstrate the feasibility of presenting activity-specific information to the user prior to entering an IMRL generation process in the SERMIS Allowance subsystem. In order to perform this function, activity data was separated into two categories: employment data and remarks. Employment data for this

project was limited to: AAI, CI, Activity name, and the Three Degree Code of engine repair. Remarks were considered to include any supplemental information on an activity that might affect an IMRL generation. For example, an activity remark on an aircraft carrier would be provide the models of aircraft comprising the airwing.

When a user begins the program, the first choice to make is the activity for which information is desired. All of the activities within the COMNAVAIRPAC SECA are arranged alphabetically, by name, and displayed for selection in groups of approximately 45. The user must determine the appropriate alphabetical range for the desired activity. Once the correct range of names is selected, all the names within that range are displayed and then the specific activity can be chosen. After making an activity choice, options are given for viewing Three Degree Codes and supported/supporting activities. The main knowledge base file, AIRPAC.KBS, will then call the remarks knowledge base file (using the VP-Expert technique known as chaining) , REMARKS.KBS and determine if there is any supplemental information to be displayed.

D. SYSTEM FILES

This prototype made use of several database files and two knowledge bases. The database files were constructed using DBase III Plus and have the "dbf" file extension associated with them. The knowledge base files were created using the VP-Expert text editing feature and are designated by a "kbs" file extension. Table E-1 provides a summary listing of the all the files used in this prototype, while Table E-2 gives a summary listing of the database file structures used by the program.

TABLE E-1. File Summary

File Name	File Type	Description
AIRPAC	KBS	Main knowledge base file
REMARKS	KBS	Contains remarks on COMNAVAIRPAC's activities
ACT_MSTR	DBF	Holds the 369 COMNAVAIRPAC activity names
AIRPAC0	DBF	Used for menu display of names from A6E Planning to HAMS 13
AIRPAC1	DBF	Used for menu display of names from HAMS 15 to HMH 463
AIRPAC2	DBF	Used for menu display of names from HMH 465 to MAG 11
AIRPAC3	DBF	Used for menu display of names from MAG 12 to NAF Misawa
AIRPAC4	DBF	Used for menu display of names from NAS Adak to USS Tripoli
AIRPAC5	DBF	Used for menu display of names from VA 22 to VFA 195
AIRPAC6	DBF	Used for menu display of names from VF 1 to VMQ 2
AIRPAC7	DBF	Used for menu display of names from VP 1 to VXE 6
3DEGREE	DBF	Contains type engines and the 3 D codes for "I" level activities

TABLE E-2. Database File Structure Summary

File Name	Number of Records	Number of Fields	Field Names & Types
Airpac0	49	3	CI: character; AAI: character; Name: character
Airpac1	49	3	CI: character; AAI: character; Name: character
Airpac2	42	3	CI: character; AAI: character; Name: character
Airpac3	54	3	CI: character; AAI: character; Name: character
Airpac4	47	3	CI: character; AAI: character; Name: character
Airpac5	56	3	CI: character; AAI: character; Name: character
Airpac6	37	3	CI: character; AAI: character; Name: character
Airpac7	35	3	CI: character; AAI: character; Name: character
Act_Mstr	369	3	CI: character; AAI: character; Name: character
3 Degree	165	3	Facility: character; Engine: character; Level: numeric

E. FUTURE RECOMMENDATIONS

This prototype only made use of the AAI, CI, and Three Degree Codes employment data elements in order to demonstrate project feasibility. Future efforts should strive to incorporate the remaining employment data elements. These elements can either be incorporated as additional fields in the current database files, or as separate files.

Ultimately, it is desired to have the remarks knowledge base file (REMARKS.KBS) contain a rule for each activity. In this project, activity information was limited to the brief examples used by the expert and parallel examples based on the author's experience. More time with the expert is required to quantify this activity information. As the information is gained, it can be added directly into the remarks knowledge base file.

APPENDIX F

RULES AND INFORMATION BASE EXAMPLES

A. OVERVIEW

The two primary methods of encoding the knowledge for this prototype were database files and knowledge base files. The database files were considered the "information bases" while the knowledge base files were comprised of the rules. This appendix provides examples of these methods that are indicative of the overall design, and addresses the testing involved with them.

B. EXAMPLES

The type of information contained in the database files was factual data regarding the activity, such as the Three Degree Code of Gas Turbine Engine Repair. Figure F-1 displays a representative sampling from the database "3Degree.dbf," which was used as one of the information bases for the prototype.

Facility	Engine	Level
•	•	•
•	•	•
•	•	•
NAS_Alameda	J52_P_6B-8B-408	1
NAS_Alameda	J57_P_10	1
NAS_Alameda	TF_41_A-2B-2C-402D	3
NAS_Alameda	T58_GE_8F-10	2
NAS_Alameda	T64_GE_6	2
NAS_Agana	J57_P_10	1
NAS_Agana	T56_A_10-14	3
NAF_Atsugi	T58_GE_8F-10	1
NAF_Atsugi	T56_A_14	2
•	•	•
•	•	•
•	•	•

Figure F-1. Information Base Example

In order for the expert system to use this knowledge, a rule would have to be satisfied in the knowledge base that would contain instructions to retrieve this data from the information base. An example of such a rule is provided in Figure F-2.

```
RULE 3Degree

IF 3Deg_menu = yes
THEN
        DISPLAY "Checking 3 degree codes for {the_activity}"
        GET the_activity = facility, B:3Degree, ALL
        FIND message
```

Figure F-2. Rule Accessing an Information Base

The rule "3Degree" in Figure F-2 picks up just after the user has been presented an option to view the Three Degree Codes. If the user responded "yes" to this option, then the rule premise in this example has been satisfied. The program then presents a message to the user, via the "display" statement, that it is checking for the codes. It should be noted here that the variable name, "the_activity" has been assigned to the activity name the user selected at the beginning of the consultation. The "GET" clause is the next instruction the program encounters. This clause is telling the system to transfer values from a database to the rule base. In this case, it is going to the "3Degree" database file on the "B" drive for the values. The "GET" clause contains three arguments. The first argument is "the_activity = facility" and it is telling the expert system to select the next record from the database whose "facility" field matches the value of "the_activity." The next argument in this example is "B:3Degree" and this is simply the path and filename for the database to access. The third argument, "ALL," mears that all the field values of the selected record should be transferred.

It was mentioned that the "GET" clause retrieves records one at a time from the database. In the case of multiple occurrences, as with NAS Alameda in Figure F-1, a loop condition had to be constructed in order to retrieve all values for an activity. VP-Expert does this through the use of the "WHILEKNOWN" clause. Figure F-3 provides an example where the rule from Figure F-2 has been modified by an addition of a "WHILEKNOWN" condition.

```
RULE 1
IF 3Deg_menu = yes
THEN
  WHILEKNOWN facility
      GET the_activity = facility, B:3Degree, ALL
      FIND message
      RESET message
  END;

RULE 2
IF facility <> UNKNOWN
THEN
      message = displayed
      DISPLAY "For the {engine} engine, this is a level {level} facility"
ELSE
      DISPLAY "This completes the list. Press any key to continue...~";
```

Figure F-3. "WHILEKNOWN" Example

Rule 1 in Figure F-3 contains the "WHILEKNOWN" condition and the loop runs between the "WHILEKNOWN" and "END" keywords. As long as the variable named in the "WHLEKNOWN" clause has a known value, the loop will execute. In this example the variable is "facility," so as long as a record is found in the "3Degree" database file in which "the_activity" matches a "facility" field value, the value of "facility" will be known

and the loop will continue to execute. The next clause in this loop instructs the inference engine to "find" a value for "message." The first place the inference engine looks is in the rule conclusions. Rule 2 in Figure F-3 contains the variable "message" in its rule conclusion. After finding this variable, the inference engine will check to see if the rule premise is valid. In this example, the premise can be translated to "if facility does not equal unknown," that is, if the facility has a known value. Assuming a value for facility is known, the system will display a message reflecting the values found from the database.

In summary, this example set up a loop condition (Rule 1 in Figure F-3) in order to obtain multiple values from a database depicted in Figure F-1. As long as a value was for "the_activity" was found to match a value for "facility" the loop would execute and make Rule 2 display the results. The next step was to run test cases to validate this aspect of the design.

C. TESTING

An activity was selected for which there were multiple occurrences in the "3Degree" database file. NAS Alameda was one such test case and it will be used here for purposes of illustration. Once NAS Alameda was selected from the menu as the activity for which information was desired, "the_activity" was assigned the value "NAS_Alameda." If the user desired Three Degree codes, Rule 1 in Figure F-3 would be valid and the inference engine would then go to the database shown in Figure F-1 and select the first record where "the_activity," that is "NAS_Alameda," was the same as the value in the "facility" field. The program would then return all the values for this record (Facility, engine, and level) back to the rule base. The inference engine was then instructed to find a value for "message" and went to Rule 2 in Figure F-3. Checking the premise, it found the value of facility was known, therefore the premise is true and the conclusion was executed by

displaying on screen the "engine" and "level" values brought over from database file. The program then continues loop execution since "facility" is still known. After the fifth iteration no more records will be found that contain NAS_Alameda in the "facility" field. Consequently, the loop variable, "facility" now becomes unknown satisfying the exit criteria, and the "else" portion of Rule 2 is also satisfied causing the complete message to be displayed. Figure F-4 shows how the output from this test case would appear on the screen.

```
For the J52 P 6B-8B-408 engine, this is a level 1 facility.
For the J57 P 10 engine, this is a level 1 facility.
For the TF 41 A-2B-2C-402D engine, this is a level 3 facility.
For the T58 GE 8F-10 engine, this is a level 2 facility.
For the T64 GE 6 engine, this is a level 2 facility.
This completes the list. Press any key to continue...
```

Figure F-4. Screen Display of Three Degree Codes.

One important problem the test cases uncovered with this design was the satisfying of the record selection criteria from the database, that is, the "GET the_activity = facility" portion of the "GET" clause. The value for "the_activity" had to match exactly the value for "facility." The variable, "the_activity" is assigned a value based on an opening menu display of the activity names from one of the databases "Airpac0" through "Airpac7." The value for "facility" was taken from the field of the same name of the database file "3Degree." The tests revealed that in some cases, even though the names were spelled correctly in the two separate files, an extra space or the absence of an underscore would cause the selection criteria to be invalid. Particular attention had to be given to the names in both files to ensure satisfying the selection criteria.

REFERENCES

[Alavi 84] Alavi, M. and Napier, H. A., "An Experiment in Applying the Adaptive Design Approach to DSS Development," *Information and Management,* 7, 1984, pp 21-28, reprinted in *Decision Support Systems,* Ralph Sprague and Hugh Watson, [eds.], Prentice-Hall, 1986.

[Blanning 84] Blanning, R.W., "Issues in the Design of Expert Systems for Management," *AFIPS Proceedings,* (National Computer Conference '84), 1984, 489-495.

[Boehm 81] Boehm, B., *Software Engineering Economics,* Prentice-Hall, 1981, p. 37, quoted in Pressman, Roger, S., *Software Engineering, A Practitioner's Approach,* McGraw-Hill, New York, 1987 p. 149.

[Boose 86] Boose, John, H., *Expertise Transfer for Expert System Design*, Elsevier Publishing Company Inc., New York, 1986, 312 pp.

[Buchanan 82] Buchanan, B.G., "New Research on Expert Systems," 269-299, printed in *Machine Intelligence,* Hayes, J.E., Michie, D., Pao, Y-H., [eds.], Halsted Press, 1982.

[Cooper 86] Cooper, Philip, "Expert Systems in Management Science," *Future Generations Computer Systems,* 2(4), December, 1986, 217-223.

[Feigenbaum 84] Feigenbaum, Edward, A., "The Art of Artificial Intelligence - Themes and Case Studies of Knowledge Engineering," reprinted in *Artificial Intelligence* , O. Firschein [ed.], AFIPS Press, Reston, VA , 1984.

[Ford 85] Ford, F. Nelson, "Decision Support Systems and Expert Systems: A Comparison," *Information and Management,* 8, 1985, 21-26.

[Harmon 85] Harmon, Paul, and King, David, *Expert Systems,* John Wiley & Sons, New York, 1985, 283 pp.

[Hart 86] Hart, A., *Knowledge Acquisition for Expert Systems,* McGraw-Hill, New York, 1986, 180 pp.

[Hayes-Roth 83] Hayes-Roth, Frederick, Waterman, Donald A. and Lenat, Douglas, B., ed.,*Building Expert Systems,* Addison-Wesley, Reading, MA, 1983, 444 pp.

[Henderson 87] Henderson, John C., "Finding Synergy between Decision Support Systems and Expert Systems Research," *Decision Sciences*, 18, 1987, 333-349.

[Heng 87] Heng, M.S.H., "Why Evolutionary Development of Expert Systems Appears to Work," *Future Generations Computer Systems,* 3, 1987, 103-109.

[Hogue 84] Hogue, J. T., and Watson, Hugh J., "Current Practices in the Development of Decision Support Systems," *Proceedings from the International Conference on Information Systems,* 1984, reprinted in *Decision Support Systems,* Ralph Sprague and Hugh Watson, [eds.], Prentice-Hall, 1986.

[Hu 87] Hu, David, *Programmer's Reference Guide to Expert Systems,* Howard W. Sams & Company, Indianapolis, IN, 1987.

[Keen 78] Keen, P.G. and Scott Morton, M.S., *Decision Support Systems: An Organizational Perspective,* Addison-Wesley, Reading, MA, 1978, p.74.

[Keen 81] Keen, P. G., "Value Analysis: Justifying Decision Support Systems," *MIS Quarterly,* 5(1), March, 1981, reprinted in *Decision Support Systems,* Ralph Sprague and Hugh Watson, [eds.], Prentice-Hall, 1986.

[Luconi 86] Luconi, Fred, L., Malone, Thomas, W., and Scott Morton, Michael, S., "Expert Systems: The Next Challenge for Managers," *Sloan Management Review,* Summer 1986, 3-14.

[Michie 82] Michie, Donald, ed., *Introductory Readings in Expert Systems,* Gordon and Breach, New York, 1982, 237 pp.

[Mishkoff 86] Mishkoff, Henry, C., *Understanding Artificial Intelligence,* Texas Instruments Incorporated, Dallas, TX, 1986, p.3-1.

[NAVAIRINST 87] Naval Air Systems Command Headquarters , *NAVAIR INSTRUCTION 13650.1B*, 1987, Encl (5).

[O'Leary 87] O'Leary, D.E., "Validation of Expert Systems-With Applications to Auditing and Accounting Expert Systems," *Decision Sciences*, 18, 1987, 468-486.

[Olson 87] Olson, J. R., and Reuter, H.H., "Extracting Expertise from experts: Methods for Knowledge Acquisition," *Expert Systems,* 4(3), August 1987, 152-168.

[Ow 87] Ow, P.S. and Smith, S.F., "Two Design Principles for Knowledge-Based Systems," *Decision Sciences,* 18, 1987, p. 430.

[Philips 88] Phillips, Jack S., and Sanders, Perry, "First Steps in Prototyping," *AI Expert,* 3(5), May 1988, 64-68.

[Pressman 87] Pressman, Roger, S., *Software Engineering, A Practitioner's Approach,* McGraw-Hill, New York, 1987 p. 149.

[SERMIS 86] Naval Aviation Logisitics Center, Management Systems Development Directorate, *Support Equipment Resources Management Information System Users Manual,* September 86.

[Sprague 80] Sprague, Ralph, H., "A Framework for the Development of Decision Support Systems", *MIS Quarterly,* 4(4), June, 1980, reprinted in *Decision Support Systems,* Ralph Sprague and Hugh Watson, [eds.], Prentice-Hall, 1986.

[Sprague 82] Sprague, Ralph, H. and Carlson, Eric, D., *Building Effective Decision Support Systems,* Prentice-Hall Inc. Englewood Cliffs, NJ, 1982, 329 pp.

[Tou 85] Tou, Julius, T., "Knowledge Engineering Revisited," *Computer & Information Sciences,* 14(3), June 1985, p.123.

[Turban 86] Turban, Efraim, and Watkins, Paul R., "Integrating Expert Systems and Decision Support Systems," *MIS Quarterly,* June 1986, 121-136.

[Vedder 87] Vedder, R. G. and Mason, R.O., "An Expert System Application for Decision Support in Law Enforcement," *Decision Sciences,* 18, 1987, 400-414.

[VP-Expert 87] Paperback Software International, *VP-Expert*, 1987, 1.1-10.3.

[Waterman 86] Waterman, Donald, A., *A Guide to Expert Systems*, Addison-Wesley Publishing Company Inc., Reading, MA, 1986, 419 pp.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5102	2
3.	Director, Information Systems (OP-945) Office of the Chief of Naval Operations Navy Department Washington, D.C. 20350-2000	1
4.	Naval Aviation Maintenance Office Attn: John Gray Naval Air Station Patuxent River, Maryland 20670-5446	2