THE DEVELOPMENT OF A GENERIC EXPERT SYSTEM FOR ENTOMOLOGICAL APPLICATIONS

By

GEETA V. GUDAVALLI

Bachelor of Science in Arts and Sciences

Oklahoma State University

Stillwater, Oklahoma

1984

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 1987

THE DEVELOPMENT OF A GENERIC EXPERT SYSTEM

FOR ENTOMOLOGICAL APPLICATIONS

Thesis Approved:

M J Folk
Thesis Adviser

Garrett Cupery

Don Cyrus

Norman N. Durham
Dean of the Graduate College

PREFACE

I wish to express my sincere gratitude to the individuals who assisted me in this project. In particular, I wish to express my sincere appreciation to my major adviser , Dr. Michael J. Folk , for his valuable guidance and advice throughout my graduate program. I wish to express my gratitude to Dr. Gerrit Cuperus for suggesting this project and for providing me with financial support throughout my graduate studies. I would like to thank Dr. K. M. George for his helpful suggestions.

I also would like to acknowledge Dr. Jerry Young and Dr. Linda wilson for their many discussions during the completion of my thesis project. In addition, I would like to express my appreciation to the faculty and the staff in the Departments of Entomology and Computer Science for their encouragement and friendship throughout my graduate studies.

Finally , I would like to thank my husband Ram for his moral support and his encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

In entomology, computers are becoming more popular as teaching tools and information managers. Computers are used in entomology mostly for word processing, data acquisition, or data analysis. Knowledge-based systems often can be used in research, in agricultural decision making, and in planning. These knowledge-based systems are popular only when the decisions are made by the humans. Knowledge-based systems, also known as expert systems, have been developed and used in several areas including: Industry, Medicine, Engineering and Chemistry. But there has been very little attempt to develop expert systems either in entomology or in agriculture. The reason for this is not lack of the problems that can be solved by expert systems, but probably the lack of exposure to expert systems and the technology.

One of the applications of artificial intelligence is expert systems and only recently artificial intelligence has advanced to a point where AI projects are accomplishing practical results. Many of these results can be used in the development of expert systems. One important application of expert systems is decision making systems. An expert system contains a knowledge base and an inference mechanism that is

able to conduct formalized reasoning. By relating the contents of the knowledge base to the information provided by an user's answers to system-formulated questions, the system infers the most recommended action in any particular situation.

Problem Statement

There are a number of insects that infest different crops in Southwestern Oklahoma. These infestation problems are characterized by the variety of crop planted and the planting date of the crop. The crop might already be growing and the farmer needs to know whether or not to apply an insecticide and what type of insecticide to apply. The decision is complicated by many other factors and depends mainly on the time of occurrence of a particular insect. The problems include quantification of the insects, and the uncertainty in their future population growth. If the farmers fail to treat the field in right time, it could result in considerable or sometimes total loss of crop. Early use of insecticides in the year could result in elimination of natural enemies. The task of this proposed problem is to develop a generic expert system to provide recommendations for control of these insects. The use of this system is demonstrated by applying it to a problem involving the control of insects in cotton crops in Southwestern Oklahoma.

The basic characteristics that make the proposed problem suitable for the application of AI techniques are as

follows:

The problem is complex enough to need representation of an expert knowledge, but it is also simple enough to be of reasonable size. When a recommendation is given to control an insect, consideration is also given to the potential for other pests in the field. Many of these considerations are based on experience and can be used in a knowledge base of the proposed system. Most of the knowledge that is obtained from domain experts can be represented exactly in a knowledge base in the form of production rules.

CHAPTER II

OVERVIEW OF EXPERT SYSTEMS

Introduction to Artificial Intelligence

Artificial Intelligence is a branch of computer science that is concerned with making computers behave in a way that would be considered intelligent in a human being [17]. The word intelligence in AI has been identified in various ways by various people. Intelligence means the ability to gather information, use that information where needed, make connections between various pieces of information and reach a conclusion about something [17].

Computers are well suited to store information and systems can be built to make connections between various pieces of information. These two aspects of computer operation are considered to be necessary parts of artificial intelligence. The following list gives a summary of problems that fall within the scope of artificial intelligence.

1. Expert and Knowledge-based Systems
2. Natural Language Processing
3. Perception
4. Game Playing
5. Pattern Recognition

Definition of an Expert System

Edward Feigenbaum, a pioneer in expert systems, [5] stateS:

> **An expert system is 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 necessary to perform at such a level, plus the inference procedures used, can be thought of as a model of the expertise of the best practitioners of the field.**

The knowledge of an expert system consists of facts and heuristics. Facts are short-term information that can be changed rapidly during the course of a consultation. They constitute a body of information that is widely shared and generally agreed upon by experts in the field. Heuristics are long-term information about how to generate new facts or hypotheses from the present knowledge or information. They are little discussed rules of plausible reasoning that characterize expert-level decision making in the field [5].

Several factors should be considered in developing an expert system:

i. The expert system should be able to make a tentative conclusion when enough data is not available concerning the problem in order to make a decision.

ii. The expert system should be able to deal with data that is incorrect or inaccurate.

iii. The expert system should be able to document and explain line of reasoning used that led to the system's recommendations.

iv. The expert system should be able to deal with ideas that are of variable or unknown certainty.

v. The problem should be complex enough to need representation of expert knowledge, but also simple enough to be of reasonable size.

Basic Components of an Expert System

Basically, every expert system contains a knowledge base and an inference engine. However, other components may be present to enhance the scope of the system. Figure 1 shows the structure of a typical expert system. The knowledge base is a dependency network of rules, facts and heuristic knowledge about a subject encoded so that relationships among the elements are obtained. It contains all of the information that is specific to the particular task that is needed to draw a conclusion. The inference engine determines and controls the sequence of logical steps that will probably lead to the solution of a user's problem. The inference engine uses the information in the knowledge base.

The knowledge base is generally developed through an interview process involving a knowledge engineer and domain experts. Most expert systems represent the information in the knowledge-base in the form of production rules and facts. A production rule is an antecedent-predicate clause or if-then statement. Each rule contains a left side and a right side. An example of a typical rule in an expert system is as follows:

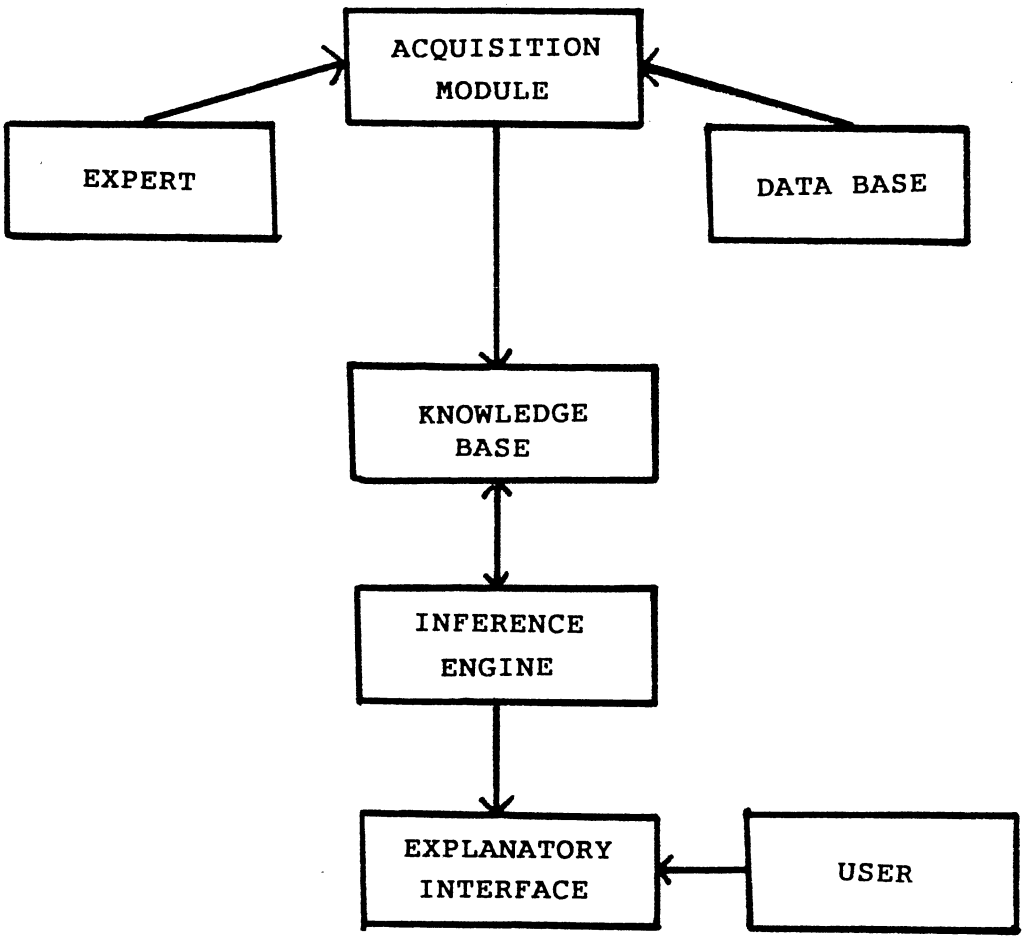

Figure 1. Structure of a typical expert system

```
Rule R1:  if condition 1
              condition 2
               .
              condition n
          then action 1
               action 2
                .
               action n.
```

The symbols RULE, IF and THEN do not contain any logical meaning and they are included only to make the representation more readable to the user. The left side of the production rule contains a description of the conditions to be tested in the knowledge base; the right side contains actions to be carried out in the event that the conditions have been satisfied by the left side. These actions may include updating the values of variables, calling another subroutine or giving a recommended decision. When domain knowledge is stored as production rules in knowledge base, the knowledge-base is usually known as the rule-base and the inference engine as the rule interpreter [5].

The inference engine is supervisor of the system's reasoning process. In a rule-based expert system, the reasoning process is obtained by proceeding from one rule to the next until all the conditions in a rule are satisfied by the current situation. Two principal control strategies are used to proceed with the reasoning process: Backward chaining and forward chaining.

Backward chaining

In backward chaining, the system has a set of initial goals, and the rules are involved in reverse order. The system begins by examining the rules in the knowledge base whose right-hand sides satisfy that goal. The system then proceeds to examine the left-hand side of the rules to see which of the goals are satisfied. This process continues until all the necessary conditions are met or some necessary conditions are contradicted.

Forward chaining

In forward chaining, the system does not start with any particular goals defined for it. The system has no initial set of production rules which establish a starting point. The initial step is to get information from the user. This information then defines a pattern of the state of the world. The inference engine searches the knowledge base for the production rules whose antecedent match that pattern. When the conditions in a rule are satisfied by the current situation, the rule is said to be triggered. The inference engine then performs the actions specified in the rule's predicate.

In general, both forms of invocation and evaluation of the production rules are valid as long as they yield the same correct conclusions. The path taken to arrive at these conclusions may differ depending on the strategy used. Forward chaining is well suited for the problems where the goals are loosely defined or where there are multiple goals

for a given problem. Backward chaining is better for problems with well defined goals. Forward chaining is said to be data driven where as backward chaining is said to be goal driven [5].

In some cases, subroutines provide information needed by the system. The subroutines can be called through execution of a rule in a knowledge base [2].

Knowledge Acquisition

Knowledge acquisition is the transfer and transformation of problem-solving expertise from some knowledge source to a program. There are several sources for knowledge acquisition including human experts, text books, and data bases. Acquiring knowledge from human experts in an expertise is known as knowledge engineering. Knowledge engineers acquire knowledge from a human expert and then embed it in an expert system. The most difficult aspect of knowledge acquisition is the initial step of helping expert structure the domain knowledge for use in solving the problem. Because the knowledge engineer has far less knowledge of the domain than does the expert, the process of acquiring knowledge and embed it into program is bound to suffer from communication problems. There are several modes of acquiring a source to a program. The source is generally a human expert, but could also be the primary sources from which the expert has gained the knowledge: journal articles or experimental data.

Knowledge Representation

There are several different ways to represent the knowledge that is acquired from an expert in a knowledge base. Frames and production rules are two of the methods to represent expert knowledge in a knowledge base [7]. A frame is a description of an object that contains slots for all of the information associated with the objects. Slots may store values. Production rules are the most popular and effective representation form for describing domain-dependent knowledge in a knowledge base. Production rules can be easily understood by domain experts but they do not provide an effective representation facility for most knowledge-system applications[3]. A great deal of success has been achieved by integrating frame and production rule languages to combine the advantages of both representation techniques.

Advantages over Conventional Programming

An expert system has several advantages over a conventional computer program in several respects. In an expert system, there is a clear distinction of general knowledge about the problem from information about the current problem, and there are methods for applying the general knowledge to the current problem [5]. In a conventional computer program, these elements are intermixed so it is difficult to modify the problem. Another difference between expert systems and conventional programs is that expert systems deal with heuristic knowledge where as conventional computer programs deal with data. Data are

facts that are obtained directly or derived by calculations. Similarly, an expert system would be able to explain how a particular conclusion was reached or why a particular line of questioning is being pursued [7].

There are a large number of expert-system development tools, also called expert system shells, available in the market now. These shells are designed for applications on conventional computers. They provide a framework for developing expert systems, in much the same way that application programs such as DbaseIII, Lotus 123, etc are used to develop data management applications. These shells differ from AI programming languages in several ways. They do not require a special purpose hardware for delivery of expert systems. These shells already contain the control mechanisms that determine how they reason to a conclusion. Different shells take different approaches in representing the knowledge and reasoning about it. Therefore, these shells are better suited to some problems than to others. Expert system shells vary in the way the user enters information into knowledge base [2].

CHAPTER III

EXPERT SYSTEMS IN USE

Overview

In this section, we will look at some examples of expert systems applications. Although many of the early expert systems were medical consultation systems, in thinking of an expert system in terms of its applications, we should look at a problem that is well defined in its scope but requires human expertise to solve. Many problems from entomology can be solved using a diagnosis oriented or decision-aid expert systems.

In entomology, expert systems are used for different applications. These include taxonomic identification and pesticide recommendations. These applications are classic in the sense that they deal with distinct problems and demonstrate how the expert knowledge can be encoded into rule-based systems for use by a non-expert. Some complex problems like farm management can be addressed using expert system techniques if the problem is broken into several modules. Each module can be developed as a rule-based system and the various modules thus can be integrated into one [10].

The various forms of intelligence that are shown by

insects are more interesting to entomologists. How does an insect identify hosts, build nests etc. There are vast quantities of data available on insect behavior which can be represented in a knowledge base. If this is done, it could greatly facilitate the use of AI techniques to study the behavior of insects [10].

In entomology, there are a few expert systems that are being developed. Comax, an expert system that has been developed, acts as an expert in cotton crop management. Another expert system, Plant/ds was developed to provide consultation on the diagnosis of soybean diseases [10].

Comax

Comax is an example of an advisory expert system that is integrated with a simulation model for daily use in farm management [10]. This expert system advises cotton growers on crop management. This is integrated with a computer model, Gossym that simulates the growth of a cotton plant [10].

Comax is a rule-based expert system that consists of a knowledge base, an inference engine, a simulation model, a weather station and other data [10]. The knowledge base consists of a set of rules and facts. The inference engine invokes the rules and facts and determines the decisions to be taken. One decision is to prepare data files for the simulation model. These data files contain necessary data to hypothesize the weather and to hypothesize the applications of water and nitrogen.

When the simulation model Gossym is called, it reads the data files prepared by the inference engine. This model simulates the growth of the cotton plant under the conditions specified in those data files. The results from Gossym are saved as new facts in the knowledge base. The final action of the inference engine is to invoke another rule which determines the day that nitrogen should be applied. This rule asserts new facts into the facts base [10].

Comax is designed to run continuously throughout the crop year on a dedicated microcomputer. Comax is written in Lisp and Gossym is written in Fortran.

Plant/ds

The program PLANT/ds is an expert system developed at the University of Illinois. The task of this system is diagnosis about soybean diseases. Decisions are made based on specific questions about the diseased crop and its environment[11].

Knowledge about soybean diseases is provided in the form of decision rules, one set of decision rules for each disease the system knows about. A rule states the symptoms associated with the disease and the importance of each symptom in diagnosis of that disease. When the consultation begins, PLANT/ds treats all of the diseases it knows about as working hypothesis, i.e any of the diseases could be the cause of the problem. Then it selects questions in such a way that the maximum number of the hypotheses will be

eliminated at each step. When all the questions relevant to the remaining hypotheses have been answered, PLANT/ds is ready to give a diagnosis. There may be more than one disease given in the diagnosis, but this is because symptoms for many diseases overlap. However, a confidence factor is given at each diagnosis [11].

CHAPTER IV

OBJECTIVES OF THE PROPOSED EXPERT SYSTEM

The purpose of this proposed problem is to develop a generic expert system for entomological applications. The following six factors that are described in Chapter I should be considered in developing an expert system.

i. The proposed system makes the best decisions it can and asks the users for information it needs in order to make better decisions. The proposed system reaches a conclusion with the information available, warning the users in the process that the conclusions may not be accurate in the same way that human experts make educated guesses with the least bit of information available to them.

ii. The proposed system deals with data that may be incorrect or inaccurate. Such data may be provided by the users.

iii. The expert system should be able to document and explain the line of reasoning used. The proposed system provides a trace back of rules that are used to reach a conclusion.

iv. The expert system should be able to deal with ideas that are of variable or unknown certainty. Human experts make decisions that sometimes they are not very sure about.

Many expert systems use a numerical scale to indicate the certainty factor of their facts and the hypotheses with which they deal.

v. The problem is complex enough to need representation of expert knowledge, but simple enough to be of reasonable size. There is no nice measure to know the complexity of the problem, but researchers in the field have indicated a need for expert system.

Basic Components of the Proposed System

The knowledge base for the proposed system is developed using IF-THEN rules. These rules represent expert judgmental reasoning and are obtained through interview processes from domain experts. This proposed system reasons by proceeding forward (i.e forward-chaining) in the knowledge base. If all the conditions in IF part of a rule are satisfied, it takes the action specified in THEN part of the rule. The action may then include printing a recommendation, may call another subroutine or may assert a new fact into the global data base. This system differs from expert systems in that it prompts the user for input and add it to the data base that contains facts, thus it may affect the scope of the rules.

The proposed system continues its reasoning and evaluates all the rules in the knowledge base. If none of the rules are satisfied, then the system may prompt the user for additional information in order to solve the particular problem.

The knowledge base for this proposed system contains IF-THEN rules for treatment of cotton crop if the crop is infested by insects. The global data base contains characteristics of insects that infest cotton crop.

Languages and Tools

The proposed system will be developed on IBM/PC AT microcomputer using Golden Common Lisp compiler. The proposed system is written in Common Lisp. Lisp is a programming language that is designed for applications artificial intelligence. Common Lisp is one of the dialects of Lisp language.

CHAPTER IV

METHODOLOGY

Knowledge Acquisition

There are a number of human experts for each crop in the field of entomology. The knowledge used in this proposed project is derived from experts for the cotton crop who have vast quantities of knowledge and experience. This knowledge is obtained through interview processes with the domain experts. General rule-of-thumb knowledge such as insect population thresholds, characteristics of the insects is obtained from these experts.

General Scheme

In this section, various modules involved in the implementation of the proposed expert system are described. To implement the system that was described in preceding chapters, an expert system with particular features is required. Figure 2 outlines the relationships between the expert, the knowledge engineer and the user as they interact with the expert system. The proposed system includes a knowledge base, an inference engine, and a user interface. The knowledge base and data bases shown in Figure 2 are unique to individual expert systems. For this proposed generic expert system, the inference engine needs access to

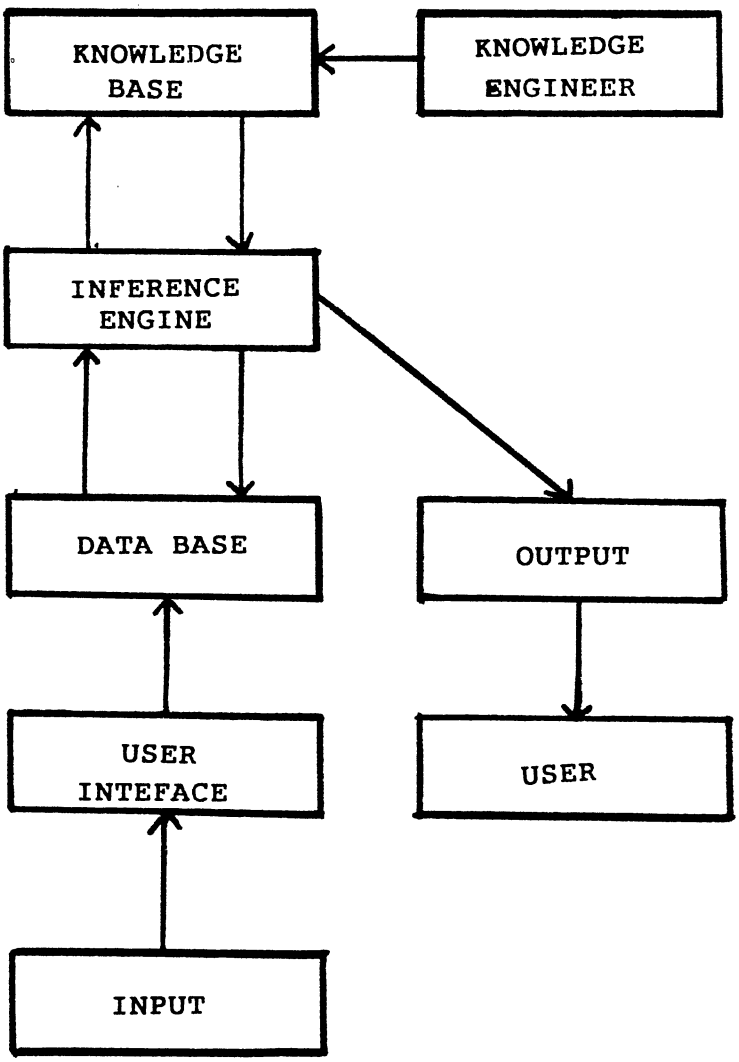

Figure 2. Representation of the process involved in developing the expert system

the data base constantly in order to read information from the data base and to write information to the data base. The proposed generic expert system provides easy access to any external procedures if needed and adds new facts to the database. A general scheme of the proposed expert system is presented below.

Knowledge Representation

For the proposed generic expert system, the static part of the knowledge base, the long-term information about insects, contains known and relevant information about insects, characteristics which help identify the insects and other information that is needed to make decisions in control of insects. This knowledge is represented in the form of rules. These rules are stated as IF-THEN condition-action pair. Figure 3a outlines english version of a rule that is used to control fleahoppers and Figure 3b presents exact representation of the rule in the knowledge base.

If the conditions in IF part are satisfied, the rule is said to be fired and the action specified in THEN part is taken. The rules use data from the global data base to make inferences and hypotheses about the current problem. The action part of the rule consists of either printing a recommendation or of evaluating a set of attached procedures. The action of evaluating the attached procedures is an escape mechanism that allows the execution of arbitrary LISP code. In some cases, where not enough information is provided to permit a decision to be made, the

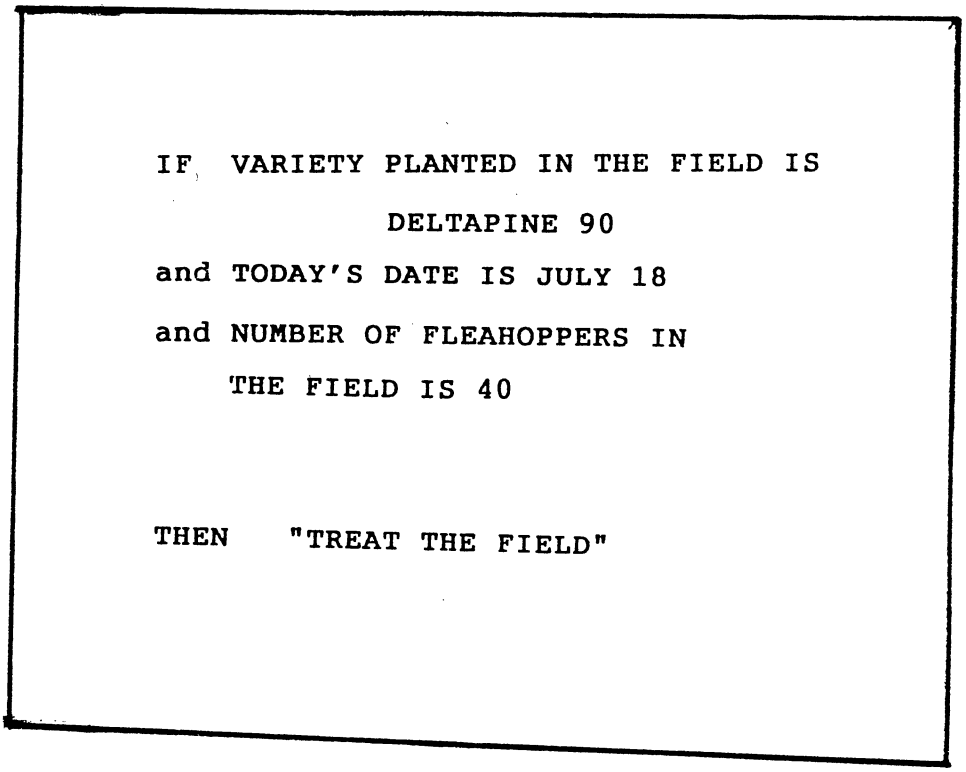

Figure 3a. Representation of a rule in English version

```
(RULE F1 (IF (AND (EQ VARIETY 1)
                  (EQ FLEANUM 40)
                  (EQ DATE 718) ) )
         (THEN ("TREAT THE FIELD")) )
```

Figure 3b. Representation of the rule in knowledge base

action part of the rule may suggest that more data is needed to make a decision.

The dynamic part of the knowledge base, that is short-term information, contains information about the current problem being considered. The user enters this information into the system in response to inquiries from the system. This information includes known characteristics on an insect, variety of a crop, planting date of the crop, insect population counts and other information that may vary from one insect to the other. All this information is stored as temporary assertions in global database. An example of such an assertion is shown below:

```
(SETQ FACTS '
      (EQ VARIETY 1)
      (EQ TODAYS-DATE 715) ))
```

All this information is stored in global database in form of a list. Each element in the list corresponds to the information that is provided by the user. For example, take the first element in the list (EQ VARIETY 1). In this, VARIETY corresponds to a variable and the value of the variable, which is 1, is provided by the user. Then this information is stored as facts in the global data base.

The Inference Engine

The inference engine provides a method used to draw a conclusion about the given problem. The basic control strategy employed by this program is forward chaining. At first, the system knows nothing about the state of the

world. The initial step is to obtain information from the user. This newly acquired knowledge provides an elementary pattern in global data base. Then the knowledge base is scanned for those rules whose antecedents match the patterns in global data base. For example, to recommend insecticide treatment for fleahopper control in cotton, the system begins by asking the user for variety of crop planted, for present date and month. The users responses are assigned to variables (i. e VARIETY 1, TODAYS-DATE 315). The inference engine searches the knowledge base for rules whose antecedents are "IF (EQ VARIETY 1) (EQ TODAYS-DATE 315)". Upon finding the above rule, the inference engine would fire that rule. This procedure is applied recursively until there are no rules to search through. If none of the rules antecedents match given assertions, then the inference engine prints a message that it may need additional information in order to give a recommendation.

Development of the Expert System

In this section, various modules involved in the implementation of the proposed generic expert system are described. The expert system is basically organized into two groups of several procedures in each. The first group contains procedures that are general to any generic expert system that is similar to proposed system. The second group contains procedures that are specific to the problem that is being tested. Figure 4 outlines the relationship between the general procedures and the specific ones. A functional

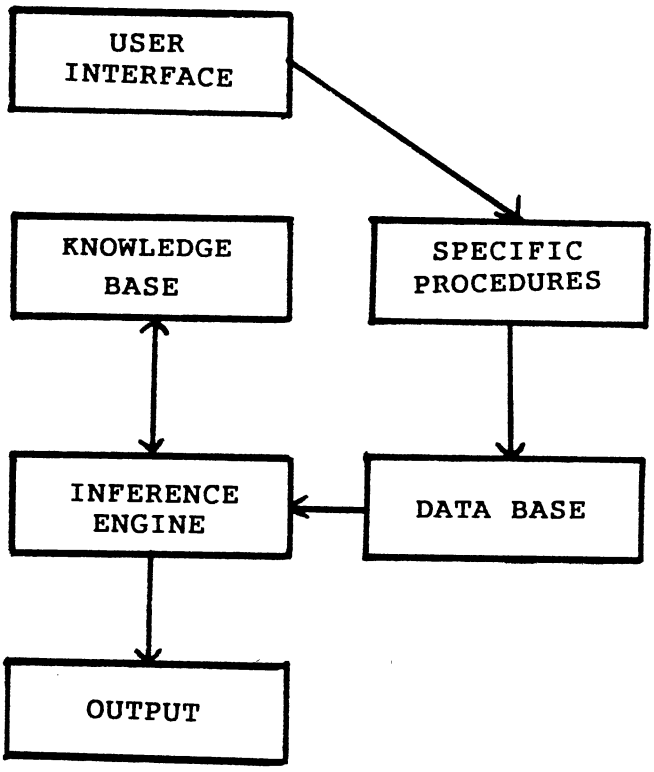

Figure 4. Representation of the relationship between general and specific procedures

description of the procedures that are general follows.

Inference-Engine

The essential purpose of this procedure is to scan through the knowledge base until there are no rules left. This is done by repeated calls to the procedure FORWARD-REASON.

Forward-Reason

This procedure is the main body of this forward-chaining program. The main purpose of this procedure is to scan down the rule list until a rule is found that works. As an initial step, this procedure initializes its local variables. One main variable in this procedure is RULES-TESTED. This variable is assigned to the value of the global variable RULES. In doing so, the value of the global variable RULES is unchanged while changes are made to the value of the local variable RULES-TESTED. The next step in this subprogram is to call the procedures FILTER-IFS and ACTION-THENS repeatedly until there are no rules left in the knowledge base or there is a rule that works. If there is a rule that works, then the necessary action is taken and the subprogram returns its control to the calling program and the execution of the calling program continues.

Filter-Ifs

The essential purpose of this procedure is to match the antecedents in a rule with temporary assertions in data file. If all the antecedents match, then the procedure sets a flag to true and returns to the calling procedure. The

initial step is to check if the list in IF part is empty. If the list is empty, then the procedure returns. If the list is not empty, then each element in the list is matched against the facts. If all the elements have a matching fact, then procedure sets the flag to true and returns to the calling procedure. If some of the elements in the list do not match, then the procedure sets the flag to false and returns to the calling procedure. This process continues until the list is empty.

Action-Thens

The essential purpose of this procedure is to perform the action that is specified in THEN part of a rule. The first step is to check if the list in THEN part is empty. If the list is empty, then the procedure returns it's control to the calling program. If the list is not empty, then the procedure takes the first element in the list and performs the action specified in the element. Then the procedure takes the second element and performs the action specified in that element. This process continues until the list is empty. The action may include displaying a recommendation or evaluating another procedure.

The following procedures are specific to the problem that is being demonstrated. Data that is necessary to make a decision is obtained from these procedures, i.e. these procedures act as user a interface. They display menus to the users and read input that is supplied by the users. Then the data is added to the data base as facts. After

creating facts, the inference engine is invoked and a solution is given. A functional description of the general procedures is given below.

Main Program

This module is the main program of the proposed expert system. In COMMON LISP, all variables are treated lexically unless the variables are arguments to a function call in which case the variables are treated as dynamic variables. For this reason, there are no variable declarations in the main program. As a first step, the main program prompts the user for his/her name and whether he/she used this program previously. If the user used this program, it displays the following menu on console.

1. Historical Temperature Information
2. Cotton Insect Control
3. Insecticide Usage for Cotton
4. Exit the Program

Depending on the user's selection from the above menu, the main program invokes corresponding subprograms. If the user chooses selection 4 in the main menu, then the program terminates. If the user has not used this program, then an welcome message appears on the screen followed by the above menu.

The internal subprograms that are called by the main program are as follows:

Insect-Control

This subprogram is called by the main program when the

user chooses selection 2 from the main menu. As a first step, this subprogram queries the user if he/she had given any previous information about the field. If so, all the necessary information about the field is read from a data file and if not, the subprogram prompts the user for a date in which the field was planted, a variety of crop planted, a closest location to the field, and present day's month and date. All of this information is assigned to different variables and finally stored in a data file. The next step of this subprogram is to prompt the user if the field is irrigated or non-irrigated. If the field is irrigated, the following menu appears on the console:

1. Thrips
2. Fleahoppers
3. Bollworm/Budworm
4. Bollweevil
5. Others

Depending on the user's Selection, the subprogram calls the procedure INFERENCE-ENGINE to evaluate necessary procedures.

Fleahopper

The essential purpose of this procedure is to create global database with temporary assertions. This subprogram is invoked when the user chooses selection 2 in the menu described previously. After creating the database with facts, this subprogram calls the procedure INFERENCE-ENGINE in order to draw conclusions about fleahopper control.

Flearoutine

This subprogram is invoked via rule F3 while evaluating the rules for fleahopper control. If variety is not 1 and present day is before July 15, this procedure is invoked to obtain further information from the user in order to draw a conclusion. The user is prompted for the number of fleahoppers per 100 terminals. This information is added to the database as another assertion and a recommendation about fleahoppers is given by issuing a call to the procedure INFERENCE-ENGINE.

Bollworm

The purpose of this routine is to query the user for information about the number of predators in the field and population level of the predators. This subprogram is invoked if the user's response to the main menu is 4. The information that is obtained from the user is again stored in database as temporary assertions. Now a recommendation regarding bollworm control is given by issuing a call to the procedure INFERENCE-ENGINE.

Pred-Collops

This subprogram is called from the procedure BOLLWORM. Upon entering this procedure, the user is prompted whether he/she has information about the number of damaged fruits or insects in the field. If the answer to this question is yes, then the procedure displays the following menu by issuing a call to the procedure DISPLAY-POP.

1. Less than 5 eggs and/or larvae/100 terminals
2. 5 to 10 larvae and/or eggs/100 terminals

3. More than 10 larvae and/or eggs/100 terminals
4. Less than 5% of the squares damaged
5. 5 to 10% damaged squares
6. More than 10% damaged squares

The user has to choose one of the six selections from the menu. All this information is stored in the database as temporary assertions and the procedure INFERENCE-ENGINE is called. If the user's response to the initial prompt is NO, then the procedure INFERENCE-ENGINE is called immediately without having the information about population level of bollworm in the field.

Collops

This subprogram is called from the procedure BOLLWORM. This procedure displays the same menu as described in the above. But this procedure is invoked from different rule from the knowledge base. Now the user has to select one of the six selections form the menu. All this information is stored in the database as temporary assertions and the procedure INFERENCE-ENGINE is called. If the user's response to the initial prompt is NO, then the procedure INFERENCE-ENGINE is called immediately without having the information about population level of bollworm in the field.

Bollweevil

The purpose of this procedure is to give necessary recommendations regarding the control of boll weevils in cotton fields. This procedure is invoked when user chooses selection 4 from insect menu in the main program. Upon entering, the procedure loads the knowledge-base and prompts the user for percentage of punctured squares in the field. If the user enters Y for this prompt, then the following menu for percentage squares appears on the screen.

1. Less than 25%
2. 15 to 25%
3. More than 25%

The user has to select one of three choices form the menu. Now the procedure INFERENCE-ENGINE is invoked and a recommendation is given. Every time, before issuing a call to the procedure INFERENCE-ENGINE, the data that is obtained

from the user is stored is a data file as temporary assertions. If the user enters N for the initial prompt, then a recommendation is displayed without displaying a menu for percentage of punctured squares.

Other

This procedure is invoked when the user selects 5 from insect menu either from INSECT-CONTROL procedure or in IRRIGTN procedure. Whether the field is irrigated or non-irrigated, this category of insects that are present in the field and the recommendations that are given for their control are likely to be same. Therefore, it is not essential to have two procedures for this particular selection. The essential purpose of this subprogram is to give decisions for control of these insects. The initial step of this procedure is to display the following menu.

1. Cabbage Loopers
2. Cotton Aphids
3. Beet Armyworms
4. Spider Mites
5. Grasshoppers
6. Cotton Leafworm

The user has to choose one of the six selections from the above menu. This information is stored as temporary fact in database and final decision is given by issuing a call to the procedure INFERENCE-ENGINE.

Irrigtn

This procedure is invoked when the user enters N for

the prompt "Is this field irrigated?" in main program. The procedure INFERENCE-ENGINE invokes this procedure through a rule from knowledge-base. The essential purpose of this procedure is to obtain necessary information about non-irrigated field from the user and invoke the procedure INFERENCE-ENGINE. The initial step is to load the knowledge base and then to prompt the user if he/she has knowledge about amount of lint/acre the field produces. If the user enters Y, then following menu appears on the screen.

1. Less than 250 pounds/acre
2. 250 pounds or more per acre

If the user chooses selection 1, then a recommendation is given immediately. If the user chooses selection 2, then the following menu appears on screen.

1. Thrips
2. Fleahoppers
3. Bollworm/Budworm
4. Boll Weevil
5. Others

Depending on the user's selection, appropriate recommendation is given by issuing a call to the procedure INFERENCE-ENGINE. If the user enters N to the initial prompt, then a recommendation is displayed immediately and the procedure returns its control to the calling program.

Add-Facts

The main purpose of this procedure is to write data that is specific to the problem to a file in the following

form.

```
(SETQ FACTS ' ((ELEMENT1)
               (ELEMENT2)))
```

For example, if the user chooses variety planted in the field is Deltapine 90 and present day is July 18 then this information is added to the data base as (EQ VARIETY 1) (EQ TODAYS-DATE 718).

Discussions

The proposed system is built by using a programming language rather than an expert system shell because of several reasons. The important reason is that this system is not designed to be a stand-alone system. Several new features may be added to the system including: integration of weather data and usage and cost effective benefits of insecticides. In this case, the system should be able to interface with external procedures that may be written in a different programming language. The above task is possible because the proposed system is built using a programming language.

The proposed system is written in Common Lisp. This language is chosen because the Lisp compiler is available on the computer where the system is being developed.

The proposed system is able to interface with programs that are written in programming language other than Lisp. This system also makes a function call to DOS in order to execute a program that is written in Fortran. No problems have been encountered with this particular task.

This system does not come under the category of expert system shell mainly because a new knowledge base can not be created by using this system. But new rules can be added or existing rules can be modified or deleted by using either a word processor or Gmacs editor that is available on Golden Common Lisp compiler. A set of instructions on how to add rules including the format of the rule is given in Appendix A.

The proposed system does not provide an explanation of line of reasoning used as in complicated systems. In complicated systems such as Mycin, users can ask "why" or "how". Then the system responds with its understanding of the question and then reports the rules it used in determining this fact. But the proposed system provides a trace back of the rules that are used to reach a conclusion. This explanation is sufficient in the case of the proposed system because this trace back provides the facts that are used to reach a conclusion in which the facts are self explanatory.

Differences from Expert Systems in Use

The proposed expert system differs from the two expert systems that are described in Chapter III in several ways.

The expert system COMAX was initially developed on a Symbolics 3670 computer and was down-loaded to personal computers. It runs under the Golden Common Lisp compiler. The proposed system was developed on a personal computer and runs on a personal computer. Comax was developed and used

for crop management applications whereas the proposed expert system was designed for entomological applications. The proposed system is also different from Comax in that it prepares data files for each user the first time when the user uses the system. The proposed system is also different from Comax that it is used as decision-aid system. The proposed system is also different from Comax that it is menu-driven, rule-based system. It displays menus to the users and asserts facts into the data base form the answers provided by the users. The proposed system is also smaller and runs faster compared to Comax. There is not enough information provided in the literature about the source code of Comax but it has knowledge base that consists of IF-THEN rules. But the format of the rules is not very clear.

The second expert system that was described in Chapter III is Plant/ds. Plant/ds was developed at the University of Illinois to provide consultation on the diagnosis of soybean diseases. It is designed to advise users about the diagnosis and decision-making regarding crop diseases and damages that occur due to insects. The proposed system is designed to provide recommendations for cotton crops. Plant/ds was developed in Plant Pathology department where as the proposed expert system was designed to use in entomological applications. From the literature available, it is not clear about the programming language and hardware that was used to develop Plant/ds.

Applications

The proposed system is designed to use as an educational tool for cotton producers in Southwestern region of Oklahoma. It will be used as on-line help service for the cotton producers. Cotton growers in this area some times face problem of deciding whether to treat the field or to wait for a certain period of time. In this case, the cotton growers can run the proposed system and should provide with a satisfactory recommendation about the particular problem. Several people have used the system and found it useful because when cotton growers have a problem with the crop, they may contact the experts in cotton or the entomologists in the area for a possible solution. In that case, the experts or the entomologists can provide a satisfactory solution by running the system. The proposed system also serves as an initial step for using artificial intelligence techniques for entomological applications.

Limitations of the Proposed Expert System

The developed generic expert system requires several file handlings in order to execute successfully. One group of files are static in a way such that they contain production rules in which the characteristics of a particular insect are encoded. These files are created individually outside the expert system. The second group of files are dynamic in nature that they contain facts about each insect. This dynamic file is created in the program and the information that is stored in this file is given by the user. If the first group of files are not present, the

program would terminate immediately. The proposed system does not explain its reasoning to the user as in complicated systems.

The present generic expert system is developed on an IBM PC/AT micro computer. Hence the developed expert system can be executed only on IBM PC/AT compatible machines. The generic expert system is developed in COMMON LISP and uses the GOLDEN COMMON LISP (GCL) compiler. The GCL compiler requires an extended memory in addition to the basic memory available on the AT computer. The developed system is not yet available as an executable file. When the expert system is available in executable file, it can be run on any IBM PC micro computers or their compatible micro computers regardless of the extended memory requirement.

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

This thesis presents the development of a generic expert system for making specific recommendations for the control of insects in various crops. The use of this system is demonstrated by applying it to a problem involving the control of insects in cotton crops in Southwestern Oklahoma. The generic expert system was developed on a micro computer using COMMON LISP language as described previously. The GOLDEN LISP compiler and the micro computer used in this project were found to be adequate to perform the tasks required at very reasonable execution speeds. At present the expert system does not support the identification of insects. The user chooses one of the insects from a menu that are usually present on the cotton crop.

The thesis presents a scheme of a generic expert system to make recommendations for insect control for cotton crop. It is possible to use this generic expert system in a daily basis in cotton fields where the generic expert system provides recommendations to the cotton growers. The explanation of line of reasoning for arriving at the decisions can be implemented with considerable programming effort. It is recommended to include a procedure to add new

rules into the knowledge base. This generic expert system can be enhanced by using computer generated graphical displays.

BIBLIOGRAPHY

[1]. Anderson J. R., Corbett A. T. and Reiser B. J., Essential Lisp. Addison-Wesley Publishing Company,1987.

[2]. Coulson R. N. and Saunders C. M., "Computer -Assigned Decision-Making as Applied to Entomology," Annual Review of Entomology, Vol 32, 1987, pp 415-437.

[3]. Fikes R. and Kehler J., "The Role of Frame-Based Representation in Reasoning," Comm. of the ACM Vol. 28, No. 9, September 1985, PP 904-920.

[4]. Gabriel R., "Lisp Tackles Data and Abstract Concepts," Computer Design, March 15, 1987, PP 78-88.

[5]. Gevarter W. B., Intelligent Machines: An Introductory Perspective of Artificial Intelligence and Robotics, Prenticeall, New Jersey, 1985.

[6]. Harmon P., and King D., Expert Systems: Artificial Intelligence in Business, John Wiley & Sons, Inc., 1985.

[7]. Hayes-Roth F., "Rule Based Systems," Comm. of the ACM. September, 1985, Vol. 28, No. 9, PP 921-932.

[8]. Hayes-Roth F., Waterman D. A. and Lenat D. B., Building Expert Systems, Addison-Wesley Publishing Company, 1983.

[9]. Latin R. X., Miles G. E. and Rettinger J. C., "Expert Systems in Plant Pathology," Plant Disease, Vol. 71, No. 10, October 1987, pp 866-872.

[10]. Lemmon H., "Comax: An Expert System for Cotton crop Management," Science, Vol. 233, July 1986, pp 29-32.

[11]. Michalski R. S., Davis J. H., Bisht V. S. and

Sinclair J. B., "Plant/ds: An Expert Consulting System for the Diagnosis of Soybean Diseases"

[12]. Nau Dana S., "Expert Computer Systems," IEEE Computer, February, 1983, pp 63-85.

[13]. Nilson N. J. and Elaine R., Artificial Intelligence. McGraw-Hill, 1983.

[14]. Pleszkun A. R., and Thozhuthaveetil M. J., "The Architecture of Lisp Machines," IEEE Computer, March 1987, PP 35-44.

[15]. Steele Jr. Guy L., Common Lisp: The Language. Digital Press

[16]. Stone N. D., Coulson R. N., Frisbie R. E. and Loh D. K., "Expert Systems in Entomology: Three Approaches to Problem Solving," Bulletin of the ESA, Fall 1986, pp 162-166.

[17]. Williamson M., Artificial Intelligence for Microcomputers, Brady Communications Company, Inc., New York, 1985.

[18]. Winston P. Henry, Artificial Intelligence. Addison-Wesley Publishing Company, 1984.

[19]. Winston P. Henry and Horn B. K. P., Lisp. Addison -Wesley Publishing Company, 1984.

[20]. Weiss S. M. and Kulikowski C. A., A Practical Guide to Designing Expert Systems. Rowman & Allanheld, 1984.

APPENDIX A

INSTRUCTIONS FOR ADDING RULES IN THE KNOWLEDGE BASE

When a new rule is added to the knowledge base,the format of the rule must be followed carefully. The format is as follows:

```
(RULE NAME (IF (AND / OR (CONDITION 1)
                         (CONDITION 2)
                              .
                              .
                         (CONDITION N) ) )
           (THEN (ACTION 1)
                 (ACTION 2)
                    .
                    .
                 (ACTION N) ) )
```

The above example shows the representation of a complete rule in knowledge base. The parentheses must match exactly as specified in the above example.

Each rule is divided into three segments. First segment is "(RULE NAME". Second segment consists of "(IF part of the rule. Third segemnt consists of " (THEN part of the rule.

The following key words" (RULE NAME" and "(IF" and "(THEN" do not provide any logical control in the program. But however, if they are present, right and left parentheses for each of them must be matched exactly as specified in the above example. These segments provide readability to the programmer.

Each IF part of the rule is divided into two parts. First part consists of a "LOGICAL OPERATOR". The logical operator can be "AND" or "OR". One of the logical operators is needed in the rules that are used in this program. If all the conditions in the IF part have to be satisfied to take the action specified in THEN part, then "(AND" is needed. If any one of the conditions can be satisfied in order to take the action in THEN part, then "(OR " is needed. The matching parenthesis for this operator should be present at the end of all the conditions in "IF" part of the rule.

The second part in "IF" segment consists of "CONDITIONS TO BE TESTED". Each condition is divided into three groups.

CONDITION ==> Arithmetic Operator | Variable | Value of Variable

Arithmetic Operators ==> one of the following (LT EQ GT GE LE)

Variable ==> Corresponds to a variable in the program

Value of Variable ==> Provided by the user or read from a file.

An example of "CONDITION" is (eq var1 1).

"THEN" part of the rule consists of either printing a message or calling another subroutine. If action is to print a recommendation or a message, then it should be written as "(FPRNT "RECOMMENDATION")". FPRNT is a subroutine to print the message.

If action is to call a subroutine, then it should be written as "(SUBROUTINE NAME PARAMETER LIST)".

The matching parenthesis for "(IF" should be present at the end of the last condition after the matching parenthesis for "(AND".

The matching parenthesis for "(THEN" should be present at end of the last action.

Each "CONDITION" and "ACTION" must be enclosed in a matching parentheses.

The final parenthesis in a rule is the matching right parenthesis for "(RULE NAME".

Rules in the knowledge base can be changed by using a word processor or by using GMACS editor that is available on GOLDEN COMMON LISP COMPILER.

VITA

Geeta Vani Gudavalli

Candidate for the Degree of

Master of Science

Thesis: THE DEVELOPMENT OF A GENERIC EXPERT SYSTEM FOR ENTOMOLOGICAL APPLICATIONS

Major Field: Computing and Information Sciences

Biographical:

Personal Data: Born in Andhra Pradesh, India, June 16, 1955, the daughter of Mrs & Mr Maniappa R. Valluripalli.

Education: Received Bachelor of Science degree in Computing and Information Sciences, Oklahoma State University,Stillwater, Oklahoma, in May 1984; Completed requirements for the Master of Science degree at Oklahoma State University,Stillwater, Oklahoma,in December, 1987.

Professional Experience: Programmer at Department of Agriculture; Oklahoma State University, Stillwater, Oklahoma, September 1983 to November 1984. Programmer/Research Assistant at Department of Entomology,Oklahoma State University, Stillwater, Oklahoma, December 1984 to June 1987.