

UNIVERSITY OF JORDAN  
FACULTY OF GRADUATE STUDIES  
INDUSTRIAL ENGINEERING DEPARTMENT

"A DIAGNOSTIC EXPERT SYSTEM  
FOR IBM PC's ( XT,AT) AND COMPATIBLES"

1111001

BY :

AMJED MAHMOUD AL-GHANIM

SUPERVISED BY :

PROF. ARUN WALVEKAR

2713  
1

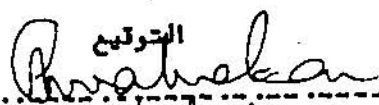
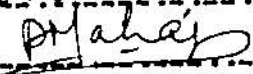
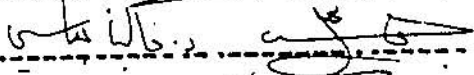

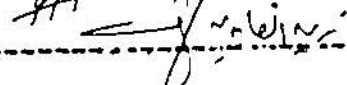
SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS  
OF THE DEGREE OF MASTER OF SCIENCE IN INDUSTRIAL ENGINEERING  
FACULTY OF GRADUATE STUDIES , UNIVERSITY OF JORDAN

August 1991

All Rights Reserved - Library of University of Jordan - Center of Thesis Deposit

This thesis was defended on 25th of August 1991,

and approved by :

التوقيع	الاسم
	11 - أ.د. ارُون وَاڤڪار
	11 - أ.د. پراكاش ماهاجان
	12 - الدكتور غالب عباسي
	12 - الدكتور ياسين حناينه
	10 - الدكتور زين العابدين تاطهويو

نوقشت هذه الرسالة بتاريخ ١٩٩١/٨/٢٥ . . . . . وأجيزت

التوقيع  
P. Mahajan

Dr. Mahajan  
Dr. Mahajan  
Dr. Mahajan

الاسم

- ١- أ.د. لرون وليفوكار
- ١١- أ.د. براكاش ماهاجان
- ١٢- الدكتور غالب عباسي
- ١٣- الدكتور ياسر حنانه
- ١٥- الدكتور زين العابدين طهوي

This thesis was defended on 25th of August 1991,  
and approved by :

التوقيع  
Arunekar

Mahajan

Dr. Ghalib Abbassi

Dr. Basil Hanaineh

Dr. Zein Alabedin T.

الاسم

أ.د. ارُون وَاڤوڪار

أ.د. پراكاش ماهاجان

الدكتور غالي عباسي

الدكتور باسيل حناينه

الدكتور زين العابدين طه مويد

Prof. Arun Walvekar

Prof. Prakash Mahajan

Dr. Ghalib Abbassi

Dr. Basil Hanaineh

Dr. Zein Alabedin T.

حيثيات قرار لجنة المناقشة (ان رجعت) :

محمد حيدر

التاريخ 28/8/91

### **Acknowledgment**

I would like to express my deep gratitude to my teachers in the Industrial Engineering Department for their care and help during my graduate study. Special thanks to my supervisor Prof. Arun Walvekar for his continuous guidance and encouragement that made this thesis possible. I also thank Engineer Zamil Ibrahim from the Electromechanical Department for his help during the knowledge acquisition phase.

## Table of Contents

### Chapter One : Introduction

	page #
	-----
1.1 - General .....	1
1.2 - Artificial Intelligence .....	1
1.3 - Areas of Artificial Intelligence ....	2
1.4 - Expert Systems .....	3
1.5 - Why Have Expert Systems ? .....	4
1.6 - Structure of the Expert System .....	5
1.7 - Features of Expert Systems .....	7
1.8 - Expert Systems Categories .....	8
1.9 - Thesis Background .....	9
1.10 - Thesis Objectives .....	10
1.11 - Importance of the Expert System .....	10

### Chapter Two : Literature Review

2.1 - Introduction .....	14
2.2 - Denral .....	16
2.3 - Mycin .....	17
2.4 - Prospector .....	17
2.5 - Macsyma .....	18
2.6 - Isis .....	18
2.7 - XSEL,R1/XCON .....	18
2.8 - The Mud System .....	19
2.9 - FSAS .....	19
2.10 - HEATEX .....	20
2.11 - EXPERTEXT .....	21
2.12 - UNIK-FCST .....	21

## Chapter Three : Expert System Development

3.1	- Fault Diagnosis .....	23
3.1.1	- What is Diagnosis ? .....	23
3.1.2	- Why is Diagnosis Difficult ? ..	23
3.1.3	- How Do People Diagnose Systems?	24
3.2	- System Development .....	25
3.3	- Identification .....	26
3.4	- Knowledge Acquisition .....	27
3.5	- Conceptualization .....	28
3.6	- Formalization .....	36
3.6.1	- Frames .....	36
3.6.2	- Semantic Networks .....	38
3.6.3	- Production Rules .....	39
3.6.4	- Knowledge Representation Scheme used in this Research .....	40
3.7	- Implementation .....	41
3.7.1	- Implementation Language .....	41
3.7.2	- Database Structure .....	42
3.7.3	- Inference Process .....	45
3.7.4	- Explanation and Learning .....	46
<b>Chapter Four : A Case Study .....</b>		<b>48</b>
4.1	- General .....	48
4.2	- Computer Printout for a Preliminary Diagnostic Session .....	51
4.3	- Computer Printout for a Deep Diagnostic Session .....	54
4.4	- Decision-Making Process .....	57

4.5	- A Human Expert-Customer Diagnostic	
	Dialog .....	60
<b>Chapter Five</b>	<b>: Conclusions and Recommendations</b> .....	61
<b>Appendix A</b>	<b>: Diagnostic Flowcharts</b> .....	63
<b>Appendix B</b>	<b>: Diagnostic Rules</b> .....	74
<b>Appendix C</b>	<b>: Program Source Code</b> .....	79
<b>Glossary</b>	<b>: List of Key Words</b> .....	95
<b>References</b>	<b>: .....</b>	98



## مُلخَص

تبنى أنظمة الخبرة لتقوم بحل مسائل مُحدّدة في مجال واضح دقيق التعريف ، وذلك باستخدام المعرفة التي تمتلكها هذه الانظمة حول ذلك المجال الدقيق . يوجد لهذه الانظمة فوائد كثيرة تقدمها عند استخدامها في التطبيقات الهندسية والتعليمية وغيرها . ففي المجالات الهندسية تُستخدم هذه الانظمة لحل مسائل التصميم ، التحكم ، المراقبة ، التخطيط ، والتشخيص وغيرها .

في هذا البحث تم بناء نظم خبرة مُحوسَب لحل مسألة تشخيص الاعطال في اجهزة الحسبات الشخصية من نوع IBM والمتوافقة معها . إن المعرفة التي يمتلكها هذا النظم تعتمد على الخبرة الفنية للأخوة من مهندسي صيانة الحسبات الشخصية ، وليس على اللبني لاسلسية التي صممت هذه الحسبات بموجبها . يقوم هذا النظم ، عند تشغيله في شركة صيانة الحسبات الشخصية ، بمهام لاربية وفنية . كذلك يمكن استخدام هذا النظم بشكل مُنفرد ليؤدي وظائف فنية تشخيصية .

تمر عملية بناء نظم الخبرة في عدة مراحل متصلة ومتداخلة وهي : الحصول على المعلومات ، صياغة المعلومات ، تنفيذ هذه المعلومات ببرنامج مُحوسب ، ثم أخيرا اختبار هذا البرنامج . لقد تم اتباع أسلوب تحليل النظم المهيكل أثناء كل مرحلة من مراحل بناء النظم . كذلك تم اقتراح وتنفيذ خوارزما معينة من أجل الحصول على المعلومات بطريقة منظمة وفعالة . لقد تم تخزين المعلومات من خلال تمثيلها بأسلوبين مختلفين هما : أسلوب لأطر وأسلوب القواعد . كذلك تم بناء قلب نظم الخبرة ، حول أربعة اجزاء فرعية وهي : جامع البيانات ، المُترجم ، العكسج ، والمُجنول . يحتوي هذا النظم على جزء خاص يقوم بتعليل القرارات التي يتخذها النظم عند حل كل مسألة معينة .

يوفر هذا النظام الامكانية لتشغيله من قبل المستخدمين على اختلاف  
مستوياتهم ، وذلك باتباع الأسلوب التحلوري مع المستخدم ، من خلال شاشات  
سهلة الاستعمال .

## Abstract

Expert systems are designed to capture the knowledge of human domain experts and apply it to solving specific problems. Such systems are already offering substantial benefits in engineering applications. They are used in design, control, prediction, planning, diagnosis, and other engineering areas.

In this research, a diagnostic computer expert system is developed for IBM PC's and compatibles. This system is a rule-based system, and suitable to operate in computer maintenance firms where it performs technical and managerial tasks. Nevertheless, the system can also be on 'stand-alone' basis to provide two types of technical consultations : Preliminary diagnosis, and Deep diagnosis.

The system development process passed through the conceptualization, formalization, implementation, and testing. Structured System Analysis (SSA) approach was applied throughout each phase. A heuristic algorithm for systematic knowledge acquisition was proposed. Mixed knowledge representation scheme was adopted using frames and production rules. The inference engine, the heart of the system, was built around the following modules : 1-data collector, 2- interpreter, 3- scanner, and 4- scheduler. The system incorporated a reasoning facility that provides decision-explanation based on experience.

The system was developed to be used by skilled computer users as well as normal operators. Therefor, it has a friendly interface with the user, through employing window menus, guiding messages, and Y/N questions.

## CHAPTER ONE

### INTRODUCTION

#### 1.1 General

The first computer revolution, which started in the mid-40's, has focused on numerical calculations. Computers have been very powerful tools for making complex and time-consuming arithmetics ; activities that were assumed to require human intelligence. Since then , we are so accustomed to calculating machines that we consider their functions to be repetitive and mechanical. Recently , a fifth computer revolution is taking place making a transition to a new era in which computers will be able to reason with knowledge. This computer age is characterized by making computers emulate human thinking. Artificial Intelligence , that part of computer science concerned with developing intelligent computer systems , has been making new areas of research since the 1960's.

#### 1.2 Artificial Intelligence

Artificial Intelligence (AI) is a term that is usually applied to functions performed by machines that would require normally some properties of human intelligence. AI also means that part of computer science concerned with developing intelligent computer systems. This involves human-like senses and reasoning [3].

AI is not just a hardware or software technology , it involves a more comprehensive process that includes data collection , representation , manipulation as well as

computer logic. Besides advances in computer hardware and software technology, AI developments requires understanding of human senses and brain. AI systems do not operate like conventional computer programs that follow certain step-by-step procedures and algorithms. They, instead, use reasoning and decision-making processes much like the human brain way of thinking [3]. Some systems can also learn from their experience and communicate in natural languages.

### 1.3 Areas of Artificial Intelligence

There are identifiable areas of research and applications that are generally, if not universally, included in the discussion of AI. The following is a brief introduction to several AI technologies [1].

a- Expert systems : these are systems that provide decision-making capabilities for specific applications that require expert knowledge. Examples include systems that are able to diagnose diseases. Expert systems will be discussed later.

b- Natural language processing : this field is divided into two sub-fields : first, natural language understanding which investigates methods of allowing the computer to comprehend instructions given in natural language and, second, natural language generation which strives to have computers produce ordinary natural language so that people can understand computers easily.

c- Artificial senses : these are special systems developed to simulate one or more of the human sensing capabilities such

as vision, learning, speech and smell.

d- Robotics : this is an extension of the mechanical robot technology that uses adaptive control , sensing to perform physical manipulation tasks , and respond flexibly to unexpected conditions.

e- Learning systems : these systems can be considered part of the expert systems. These systems are specialized in deducing new facts from their knowledge base. Learning systems are also used in the area of intelligent computer-aided education.

The work in this thesis is to be devoted to develop of an expert system , thus the rest of this chapter will highlight the basic concepts of expert systems in an attempt to have a wider comprehension of the subject.

#### 1.4 Expert Systems

One of the AI areas that claims a large measure of responsibility for the current heightened AI awareness is expert systems; computer programs that embody human experience. According to Feigenbaum [4] , the expert system is an intelligent computer system that uses knowledge and inference procedures to solve problems that are difficult and 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 in the field. Feigenbaum also adds that the expert system's work seeks to capture expertise of a field , and translate it into working computer

programs that can offer intelligent assistance to a practitioner in the field.

It is clear from the above definition that the construction of an expert system depends ultimately on two basic processes : one involves expertise transfer, that is extracting knowledge from the sources of expertise (domain human experts, manuals), and the other process involves incorporating this knowledge into a working computer program. These two processes are known as knowledge engineering. Because knowledge is a very essential part in expert systems, they are called knowledge-based systems.

### 1.5 Why Have Expert Systems ?

The expertise that the expert system is based on to solve a certain problem consists of knowledge about a particular domain , understanding of the domain problems , and the skills at solving part of these problems. This knowledge captured in expert systems databases is not public knowledge ; that is the one found in textbooks , it is a private knowledge that is embedded in the minds of human experts. This invaluable knowledge about a certain domain comes as a result of many years of experience in that domain. This knowledge is represented , in the minds of human experts , in the form of rules-of-thumb which are usually called 'heuristics'. It is very important and essential to maintain this knowledge because of the following reasons [5] :

1. storing this knowledge in a computer's memory in a suitable format avoids the possibility of losing this knowledge as a

result of death or retirement of the experts.

2. once the computer has the knowledge , it is much faster in making decisions than the human expert.

3. the expert system raises the level of performance of the expert by incorporating the expertise of many experts in that certain domain.

4. expert systems can teach new starting experts by providing opinions on how to solve the problems.

5. unlike the human experts , expert systems can be duplicated thus allowing them to be cloned and placed on many systems and utilized by many users. This is a very important factor because it solves the problem of expert scarcity.

6. the expert system captures the expertise of valued experts , thus allowing the transfer of experience from one place to another , making it possible for the expert to retire , have a vacation or go on to other tasks that could increase the level of his knowledge.

7. the greatest benefit of expert systems is consistency and uniformity in the application of decision criteria [10].

Uniformity in applying decision criteria means success in recognizing all factors that are relevant to the decision , and weighing these factors properly in all cases. Consistency means making the same decisions given the same data in different times and places.

**1.6 The Structure of the Expert System**

Currently , there is no such thing as a 'standard' expert system. Because a variety of techniques are used to



create expert systems , they differ as widely as the programmers and the problems they are designed to solve. However , the principal components of most expert systems are following :

1. knowledge base .
2. inference engine .
3. user-interface .

The component of the expert system that contains the system's knowledge is called the knowledge base. This knowledge base contains both declarative knowledge ( facts about objects , events , and situations ) , and procedural knowledge (knowledge about the courses of actions ). The two types of knowledge may be separated or integrated depending on the technique used for knowledge representation. Most expert systems use rule-based production techniques in which declarative and procedural knowledge are integrated.

The second component of the expert system is the inference engine. Simply having access to a great deal of data does not make the expert system an expert. The expert system must know how and when to apply the appropriate knowledge , and direct the implementation of knowledge. The part of the expert system that does this job is called the control structure , rule interpreter or the inference engine. The inference engine decides which rules to apply in a certain situation , which knowledge base to invoke , and determines when an acceptable solution has been found.

Finally , the user interface is the third element of

the expert system. Even the most complicated system is worthless if the intended user can not communicate with it. The communication made between the system and the user is bi-directional , and at the lowest level the communication system should make the user able to describe the problem to the expert system, and the expert system must respond with its recommendations .

### 1.7 Features of Expert Systems

Although the characteristics of the expert system depend on its nature ; its domain , intended functions , and its working environment , there are still common features that are applicable to almost all expert systems. The following is a list of the most accepted criteria used for judging (evaluating) an expert system [1] :

1. the expert system should be useful. The program must be developed in order to meet a certain need , one for which it is recognized that assistance is needed.
2. the expert system should be usable. The program must be built in order to help the novice computer user to use the program easily.
3. the expert system must respond to simple questions. Because people with different levels of knowledge may use the system , the expert system should answer questions about points that may not be clear to all users.
4. the expert system knowledge must be easily modified. It is important that one be able to revise the knowledge base of an expert system easily to correct errors or add new information.

5. the expert system is required to give reasons that lead to the advice and recommendations it makes in order to allow the user determine whether to accept the system's recommendations.

### 1.8 Expert Systems Categories

Expert systems may be applied to any situation that normally requires human expertise. Table 1.1 [3] divides the expert systems applications into ten functional categories.

Table 1.1 : Expert system categories [3].

Category	Problem Addressed
Interpretation	inferring situation description from sensor data
Prediction	inferring likely consequences of given situation
Diagnosis	inferring system malfunctions from observation
Design	configuring objects under constraints
Planning	designing actions
Monitoring	comparing observations to plan vulnerabilities
Debugging	prescribing remedies for malfunctions
Repair	executing plans to administer a prescribed remedy
Instruction	diagnosing, debugging, and repairing student behavior
Control	interpreting, predicting, repairing, and monitoring system behavior

Diagnostic systems constitute an important category because of their responsibility for the wide spread of expert systems, and their wide range of applications. In this research, an expert system has been developed for diagnosing personal computer malfunctions. IBM/XT, AT personal computers and compatibles are taken as an illustrative vehicle. The

following sections will throw light on the background and objectives of this thesis work.

### 1.9 Thesis Background

In large computer maintenance organizations, the maintenance process is carried out in a systematic manner that consists of several interrelated tasks [6]. The maintenance process includes both technical tasks such as fault diagnosis, and equipment repair, and also managerial tasks such as scheduling, resource allocation, and inventory control. In today's automated office systems for maintenance organizations, the maintenance cycle is triggered by a customer reporting a faulty equipment, or by regular preventive maintenance. The technical staff communicates with the customer asking for more information about the conditions (symptoms) of the faulty machine. Then a rough equipment fault diagnosis is made by the computer maintenance experts. The output of this step consists of : the most-likely errors that could occur in the machine, the maintenance priority level of the machine, and the necessary spare parts and field replacable units. This diagnostic information is contained in what is called a job-ticket [6]. This job-ticket information constitutes part of the input to the managerial tasks such as allocating the necessary technical workforce for equipment repair, and scheduling the different visits of the maintenance people to the many customers reporting faulty equipments.

**399400**

A critical success factor in the whole maintenance

cycle is equipment fault diagnosis. Success in performing this task effectively, in terms of minimum time and correct technical reasoning, minimizes repair time and cost, and increases the market share of the computer company.

#### 1.10 Thesis Objectives

As discussed before, the job-ticket contains essential diagnostic information that managerial staff can use to run the business of the company. This information is provided by the human maintenance expert who deduces computer faults. This process requires an expert maintenance engineer to be always available in the maintenance department. This operation is costly, time-consuming, and inefficient. A solution to this problem could be achieved by using a computer diagnostic expert system that performs the diagnostic process.

Thus, the objective of this research is to implement an off-line expert system that is capable of diagnosing computer ( IBM/XT,AT and compatibles ) failures in an interactive user-friendly manner. The block diagram of the expert system is shown in Fig. 1.2 .

#### 1.11 Importance of the Expert System

The personal computer system is a microcomputer that consists of a monitor, a keyboard, and main system unit which contains the processor, memories, control cards, power source, and auxiliary storage units such as hard disk unit. The personal computer runs under the Disk Operating System (DOS) . The expert system provides technical assistance to computer

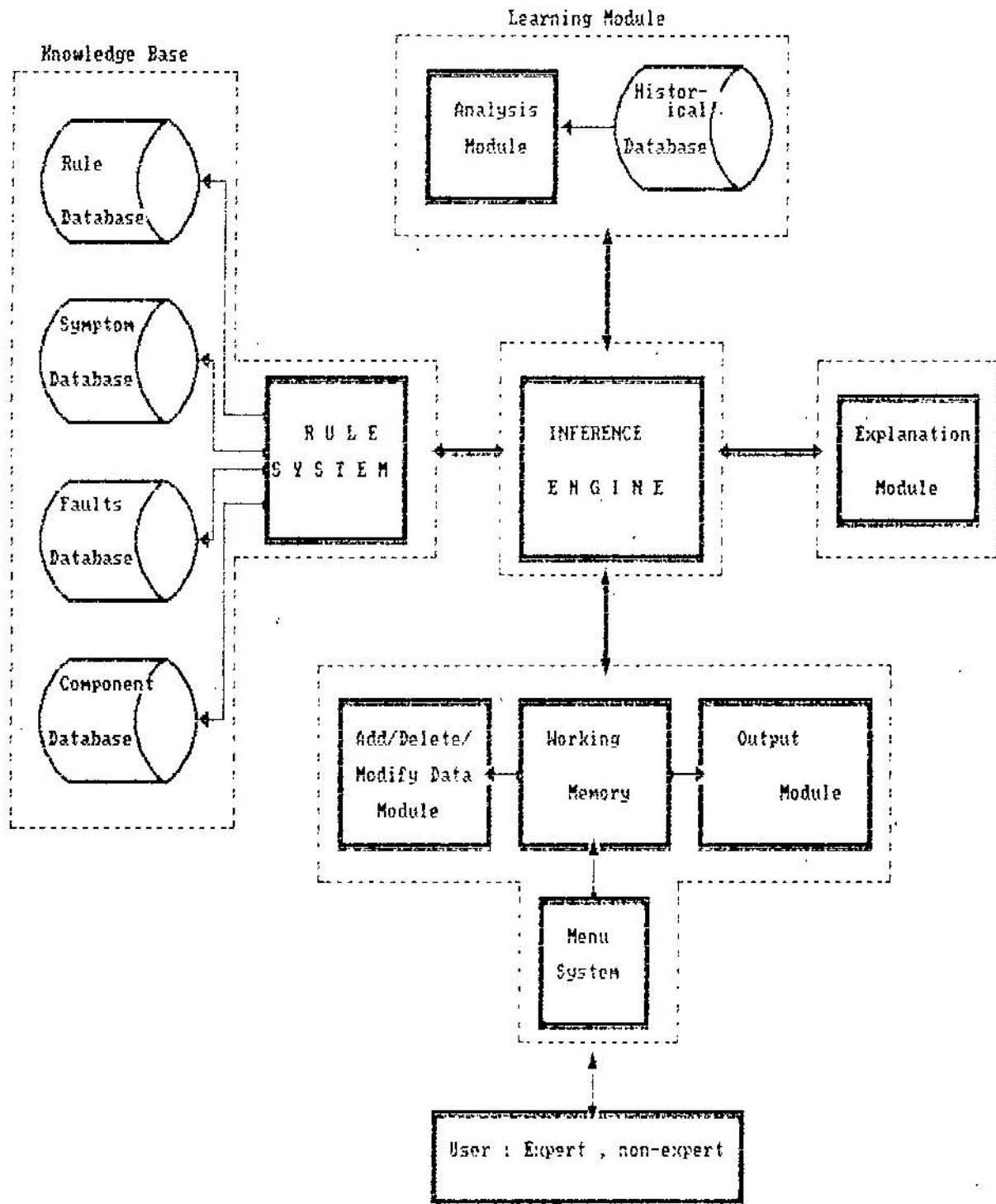


Fig. 1.2 Block diagram of the developed system

maintenance department staff through the use of two types of consultations :

1. by giving a preliminary diagnosis of equipments that are reported to be faulty by customers through making a phone call to company maintenance staff. This helps the maintenance technician to decide on the most-likely Field Replacable Units (FRU), if necessary.
2. by giving a deeper technical consultation to the maintenance technician who is working in the field. This can be done through a phone-dialogue between the system operator and the on-site maintenance technician.

Off-line expert system means that the end-user communicates with the expert system and gives the system a description about a certain computer system that is remotely located. This is different from self-testing programs that are executed by the same computer to be tested ; this is called on-line checking, which has limited capabilities and uses. The communication between the system and the user is based on interactive mode through Yes/No questions ,multilayer menus , and appropriate guiding messages.

The expert system that has been developed is not meant to be a complete system that solves all problems in this particular domain. Instead , it is a pilot system that demonstrates the implementation of the basic concepts underlying expert systems. Thus , the volume of the knowledge base and the number of problems that the system can handle are just enough to explain the system's operation.

The main questions underlying the development of expert systems are the following : knowledge acquisition, knowledge representation, control strategy, and uncertainty. All these issues are discussed and explained in chapter three.



## CHAPTER TWO

### LITERATURE REVIEW

#### 2.1 Introduction

Throughout the 30-year history of artificial intelligence, advances have been made that never emerged from research labs. However, now the impact of AI technology is starting to be felt in real-world outside the laboratory walls. This can be easily seen through the increasing interest in AI products [1]. AI university classes are filled to their capacity, many expert systems have found their ways to the market, and many AI companies and research institutes that develop expert systems and other AI-related products began to appear on a wide scale [2].

Reading through the literature of artificial intelligence in general and expert systems in particular, one can see that almost all books, articles, and papers talk about the same concepts and ideas; the requirements of developing an expert system and the application areas in which it can be used. They differ slightly in presentation methods, while keeping the main boundaries common to all of them. This may be due to the relatively short history of expert systems (25 years) [1], which includes a number of pioneers and research centers that made it possible for expert systems to gain this publicity. Table 2.1 summarizes the most important events in the history of AI, keeping in mind that the actual birth of AI was in 1956 after the famous Dartmouth Conference.

Table 2.1 Important events in the history of AI [7] .

---

1936	Turing formalizes the notion of general-purpose computer .
1945	Von Neuman conceives 'stored program' design for serial digital computer .
1946	ENIAC, the first implemented general-purpose digital computer developed .
1950	Turing describes his test for machine intelligence . Shannon speculates on the prospects for computer chess .
1955	Bernstein develops first working chess program .
1956	MaCarthy organizes the Dartmouth conference and coin the term Artificial Intelligence . Newell, Shaw, and Simon develop the LOGIC THEORIST, the first successful AI program .
1957	MaCarthy invents Lisp, the first popular AI programming language . Newell, Shaw, and Simon begin the ambitious General Problem Solver . Chomsky introduces transformational grammar to model the syntax of natural languages .
1965	Feigenbaum develops the Denral, the first expert system .
1966	Quillian invents the semantic network .
1967	Greenbalt develops MacHack, the first competent chess program .
1967	Winston's ' Learning Structural Description from Examples' pioneers machine learning . Colmerauer invents Prolog AI programming languages .
1972	MYCIN, first practical expert system to use production rules developed . Winograd completes SHRDLU natural language processing program .
1974	Minsky publishes ' A Framewok for Representing Knowledge' paper .
1975	The Lisp Machine, the first AI computer, invented at MIT . First personal computers are sold .
1982	Marr's revolutionary, comprehensive theory of vision published . Japanese ' Fifth Generation' AI project .
1986	Thinking Machine Corporation intoduces the Connection Machine . Berliner's HiTech is first chess computer to receive Senior Master title . First 32-bit microcomputers become widely available .
1988	The introduction of the 486-high speed processors
1989	Start of using software engineering tools in ES. Linking AI programs with conventional database applications.[3] Widespread of expert system shells, such as Xi-Plus.
1990	New patents: Semantic network machine for AI computers Japan, and System and method for parallel processing with mostly functional languages, USA [22].
1991	World congress on Expert Systems, Dec.16-19,1991, USA.

The rest of this chapter is discussing some of the famous already developed expert systems throwing light on the pioneers and research centers that developed them .

## 2.2 DENRAL

This is the first expert system developed in the late 1960's by a team at Stanford University. The team included Bruce Buchanan , Edward Feigenbaum , and Joshua Lederberg [1]. The system analyses unidentified chemical compounds. After interpreting the data from mass spectrometer and nuclear magnetic resonance (NMR) spectrometer , DENRAL produces topological models of molecular structure of the compound. Meta-DENRAL , one of DENRAL subprograms , is a learning program that modifies existing data and rules and discover new ones [4].

DENRAL's expertise derives from compiled hand-crafted knowledge in forms directly usable by the system [3]. It does not reason from basic principles of chemistry , so its domain is very narrow. Furthermore, its explanations are minimal . DENRAL is used in the U.S.A by a big sector of chemists .

In DENRAL the knowledge is represented as a special code for the molecular structure , and as production rules for the data-driven components and evaluators. Knowledge acquisition requires rule editing and the system has no user-interface facility .

### 2.3 MYCIN

This expert system was designed to help diagnose bacteriological blood infections. The system developed in early 1970's by Buchanan and Shortliffe , was a fruitful product of the Heuristic Programming Project at Stanford University [1]. MYCIN's expertise was derived from hand-crafted data in a limited area [3]. Like DENRAL , it has no understanding of the principles of medicine involved in its decision processes. The solution method used by MYCIN is backward-chaining (see glossary) from diagnostic hypothesis to data , under the guidance of inferential rules. The system has a convenient user interface and reasoning capability under uncertainty. MYCIN which uses production rules as a knowledge representation scheme is now used as a teaching program in medicine schools.

### 2.4 PROSPECTOR

Developed in late 1970's at Stanford Research Institute (SRI) , PROSPECTOR is an intelligent assistant in the research for mineral deposits. Having successfully predicted the discovery of a molybdenum deposits which worths \$100 millions , PROSPECTOR performance is considered to be comparable with that of the best human expert [1]. The solution method used by the system is bottom-up data interpretation. Knowledge is represented by the semantic nets, and acquired through the KAS knowledge acquisition system [3].

## 2.5 MACSYMA

Developed in the late 1960's at MIT by Carl Engleman , William Martin , and Joel Moses , MACSYMA helps solve both numeric and symbolic mathematical problems [4]. MACSYMA uses heuristic pattern matching technique to solve many types of mathematical problems , including differentiation , integration , polynomial equations , and matrices. Currently , MACSYMA is extensively used by mathematician and scientists who access it via nets on a dedicated computer at MIT .

The solution method used by the system involves no search. Instead , the results of a problem recognition phase establish an association with a specific solution method. Programmed routines are used for knowledge representation and it has a good interface facility [3].

## 2.6 ISIS

Developed in the early 1980's by the Intelligent Systems Laboratory with cooperation of Mark Fox and Stephen Smith of the Robotics Institute at Carnegie-Mellon University. ISIS is factory automation system designed to produce job-shop schedules. ISIS considers variables such as productivity goals , resource requirements , and machine preferences to construct schedules , monitor performance and avoid production bottlenecks [1].

## 2.7 XSEL,R1/XCON

These are part of the DEC series of expert systems. These are systems which configure VAX systems for DEC [18].

Inputs are customer orders and it outputs sets of diagrams depicting the spatial relationships between the components with indications of the components that are missing but necessary in order to complete the order. DEC reportedly spent several million dollars on several unsuccessful attempts at writing a configuration system with conventional techniques before achieving this successful expert system. R1/XCON is now used to configure all VAX mainframes .

## 2.8 The Mud System

Mud is a drilling fluid diagnostic and treatment consultant recently developed at Carnegie-Mellon University in cooperation with with NL Baroid [9]. Mud , now containing about 1600 rules , was developed by Cary Kahn and John McDermott in 1985.

Mud is an expert system that provides diagnostic and treatment recommendations to engineers responsible for maintaining desired properties of oil well drilling fluids. The system , implemented in OPS5 production system , is based on forward-chaining matching strategies. Mud does not take into account uncertainty factors , assuming that all data entered as well as the recognition of diagnostically significant observations is certain [9].

## 2.9 FSAS

This system, developed in 1987 [20], is a rule-based, forward chaining expert system that is used to schedule a multi-pass glassing and furnacing operation for glass-lined

vessels. Due to the nature of the manufacturing process and expected disturbances, such as rework, fabrication time of each part can not be tightly controlled. This results in the need to schedule the glassing and furnacing operations manually. This task is time-consuming and tedious. The Furnace Scheduling Advisory System (FSAS), which captures the expertise of the shop floor supervisor and uses heuristics to account for all the resource constraints, including part mix, firing temperature, and thickness difference, was developed to perform this task.

The system was developed in OPS5 and implemented on the Micro VAX II minicomputer. The system showed good results when operated on a one-month period data, as compared with the results obtained from the human expert doing the same scheduling job. The system was developed at the Standard Oil Company, Research and Development Center, Cleveland, Ohio.

## 2.10 HEATEX

This system was developed in 1989 [22] at the Department of Mechanical Engineering, Trent Polytechnic, Nottingham, UK. This system provides technical advice on how to make measurements of local convective heat transfer coefficients using a transient technique. This tool has numerous applications in assisting the design of components requiring a knowledge of heat transfer characteristics.

The system was developed using an expert system shell called 'Savoir'. HEATEX's reasoning and the rules contained in its knowledge base depends on several factors. The heat

transfer situation, type of fluid, temperature, desired accuracy, and repeatability are only part of these factors which the system takes into account.

HEATEX's advice on the equipment required includes the type the material of the test specimen, the experimental set-up, an appropriate thermal indicator, a suitable recording facility, and computer algorithm if necessary. HEATEX is a turn-key system, easy to load and consult. It employs multiple menu system and proper guiding messages.

### 2.11 EXPETTEXT

This system is an intelligent information retrieval system. This system was developed at the Department of Computer science, University of Liverpool, England, in 1990. Expertext is a system that combines the precision of expert reasoning process with the browsing capabilities afforded by hypertext. First, an intelligent hypertext is modeled with a semantic net. The semantic net formalism is then extended to Petri net formalism. Finally, a deductive inferencing ability is added to the Petri net formalism [23].

### 2.12 UNIK-FCST

This system was developed at the Korea Advanced Institute of Science and Technology, in 1990. UNified Knowledge-ForeCaST (UNIK-FCST) is a statistical expert system that solves the problem of accounting future events and other qualitative factors in the time series models. This system learns from historical judgmental adjustments through



generalization and analogy. This means that this system uses monotonic reasoning. The system uses predicate calculus for knowledge representation, and uses Prolog as an implementation language. This system takes into account certainty factor during the reasoning process [24].

## CHAPTER THREE

### EXPERT SYSTEM DEVELOPMENT

#### 3.1 Fault Diagnosis

##### 3.1.1 What is Diagnosis ?

A system needs to be diagnosed and repaired when it shows unusual behavior which can be recognized by observable symptoms. These symptoms can be observed, for example, when error indicators are lit, when there is no response to inputs, or when the system does not operate correctly (i.e. as expected). These observable symptoms have underlying reasons which can be associated with parts of the system that should be repaired so that the system functions properly. So, diagnosis is the task of identifying the underlying causes that produce the observable symptoms and correcting these causes by repairing or replacing the faulty system component.

##### 3.1.2 Why is Diagnosis Difficult ?

Diagnosis of a particular system can simply be done by enumerating all the symptoms and relating them to the underlying causes. However, this procedure is not feasible because the number of symptoms and causes are too large to be enumerated and the relations between symptoms and causes can not sometimes be well understood. In general, the diagnostic process has the following characteristics which make it a difficult job [12] :

- 1- The symptoms may have never been observed before. This is often in new systems where no historical data is available.

2- A symptom may have several causes. If each symptom is associated with a single cause, the diagnostic process will be very easy. Frequently, however, several repairs are needed to correct a single symptom and the order in which these repairs are done is very important.

3- There is no direct relation between symptoms and causes; as a set of symptoms do not have a clearly set of causes that can be corrected each time the symptoms appear. Different failures may result in the same set of symptoms, and one failure can show many symptoms.

4- Diagnosis and repair must be performed economically. One way to diagnose and repair a system is to replace the entire system with a new one. Although this procedure is simple and efficient, it is too costly. An optimal solution should take the cost of maintenance into account.

### 3.1.3 How Do People Diagnose Systems ?

Expert diagnostic systems are interactive systems that aid the diagnostic personal. So to develop an effective interactive expert system, it is necessary to understand the way people diagnose systems, and to try to implement this behavior in the developed system. Diagnosis experts usually behave according to the following measures [12] :

1- The expertise of diagnosticians is not only based on theory. The theoretical knowledge becomes 'compiled' into a set of procedures to diagnose the system that is under consideration. This means that the expert system can not rely on theory without relating it to the device being diagnosed.

2- people do not use empirical data effectively. They tend to forget what they have tried before and make tests and repairs that are not in relation to current state of the system. The expert system must have, then, a history of its diagnostic behavior.

3- Most of the time people use empirical knowledge to diagnose their systems. Sometimes, they use casual knowledge which is based on system design and configuration.

### **3.2 System Development**

The development process of an expert system takes place in a number of stages. This process which is known as Knowledge Engineering, covers two basic aspects. One is knowledge acquisition, that is, capturing the experts' knowledge, and storing it in the knowledge base according to a usable knowledge representation scheme. The other aspect is knowledge processing, which is the application of that knowledge to solving problems (knowledge manipulation). The process of expert system development passes through five interrelated phases. These phases are namely: identification, conceptualization, formalization, implementation, and testing and maintenance of the end system. All these phases are iterative in nature, and dependent on each other; the output from a certain phase may suggest modifications in other phases. These phases are discussed in the following sections.

### 3.3 Identification

The problem and its environment are determined during this phase. The way the system is to be used and the skill level of the intended users are clearly determined, as these factors affect the system design. Domain experts and knowledge engineers who are in charge of the system development are chosen. Other technical resources relating to the domain knowledge such as articles and manuals are identified and acquired. In essence, the preliminary boundaries of the problem are drawn ( i.e. what kind of problems is the system expected to solve ?, what knowledge must the expert have to solve these problems ?, and what are the final goals of the system ? etc.).

As far as this research is concerned, the problem definition and the objectives are clearly stated in sections 1.9 and 1.10. Domain experts, who are specialized in computer maintenance work, are chosen in cooperation with the Electromechanical Maintenance Department at the University of Jordan. Other technical computer maintenance manuals were acquired from a local IBM-computers dealer, and some relevant case studies were also obtained. This system is developed to offer assistance for diagnosticians in a computer maintenance company, so the system is designed to be flexible enough so that it can be integrated with other database information systems that the company could use.

### 3.4 Knowledge Acquisition

Having completed the identification phase, the next step is to acquire the relevant knowledge, and formulate it in a suitable format that is readily usable by a computer implementation language. These two tasks are referred to as conceptualization and formalization. Knowledge acquisition is a very critical and time-consuming stage of the system development life cycle. The reason for this is that experts do not have their expertise in the form of readily accessible rules that can be directly translated into knowledge base. Instead, their knowledge is 'compiled' in the form of rules-of-thumb, recognition of previously met patterns, and intuition which has been built over many years of practice and experience. Sometimes, the precise mechanism used in solving a problem and arriving at a decision is not apparent to the expert. In this case the knowledge engineer is responsible for making assumptions and having them verified by the expert. Thus, knowledge acquisition is a major factor impeding expert system development. Two basic methods are used for knowledge acquisition : Handcrafting and computer-Based Induction [13]. Both are designed to identify the basic concepts and facts used by the expert and the interrelationships between facts. In this research a systematic structured approach employing handcrafting is used to acquire knowledge.

### 3.5 Conceptualization

In this phase the specific domain knowledge is captured. Suitable concepts such as objects, functions, and problem scenarios are explored. To facilitate the process of knowledge acquisition, Structured System Analysis (SSA) is employed during this phase to enhance the expert system development method which is based on interviewing and rapid prototyping. Such methods are used because the expert system development process is very similar to conventional programming and the system analyst must have the same interviewing skills as that of the knowledge engineer. The most commonly used knowledge acquisition technique is Handcrafting [13]. Following this method, the knowledge engineer prepares by hand a representation of the knowledge used during solving a diagnosis problem. Handcrafting is employed during focused interviews that are conducted between the expert and the knowledge engineer. During these focused interviews, the expert is encouraged to diagnose a certain system fault and explore it in depth. Meanwhile, the knowledge engineer is concerned about documenting the flow of fault diagnosis tasks, and the reasons for avoiding or applying a certain action during the diagnosis activity. Discussion during interviews was conducted according to the following structured Proposed procedure :

1- As a start, the personal computer system, which is the device of interest in this research, is divided into modules with associated problems. Each module is further divided into

components that can normally be repaired. Fig.3.1 shows the main parts of the personal computer that are considered when performing maintenance.

2- During the focused interviews, handcrafted notes are made by the knowledge engineer following the verbal notes of the maintenance engineer regarding solving a particular diagnostic problem. Samples of these notes are shown in Fig. 3.2. These notes consider solving part of the problem when the system does not boot. Software and hardware problems are discussed.

3- These handcrafted notes are then formulated into flowcharts that represent the data flow and the reasoning process during problem solving. A flowchart is a good tool for representing diagnostic information as it gives a clear visual picture of the diagnostic process. These charts are called diagnostic charts.

4- After a flowchart for a particular problem is constructed, the next step is to review this flowchart, and to make any necessary corrections or modification which might change logical data flow. These changes come after arguments between the knowledge engineer and the expert about the concepts and relations that exist in the system ( component ) under consideration.

5- Steps 3 and 4 are repeated until a final sound flowchart is reached for each a certain problem. A sample flowchart is depicted in Fig. 3.3 which shows a systematic way for solving the 'system does not boot' problem, and the rest of these charts are shown in appendix A.



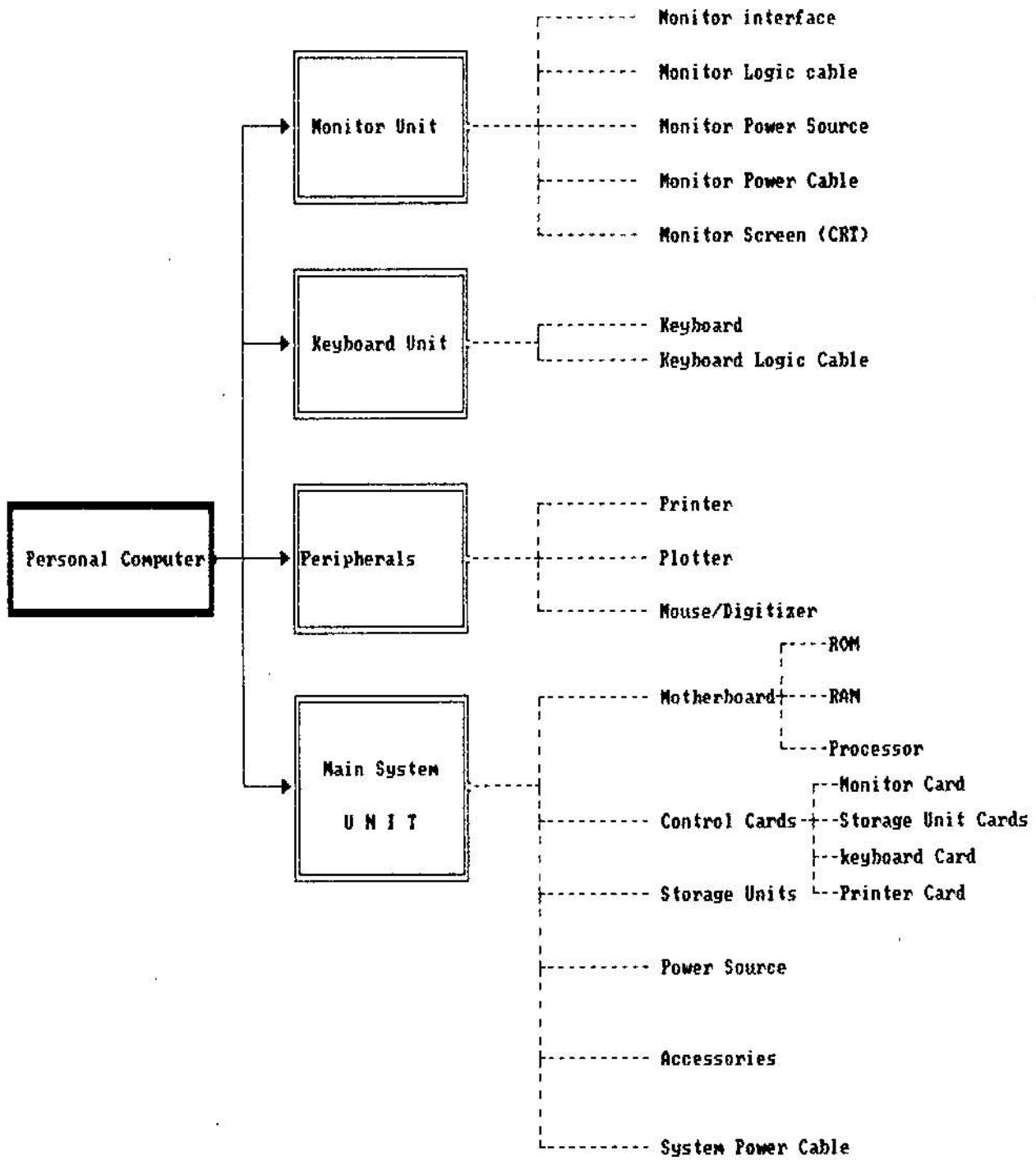


Fig. 3.1 Parts of the personal computer

March 15, 1991

### System doesn't boot :- (major Syndrome)

- error messages → most probably problem in desktop.
- after counting the memory, no response

No system disk disk

### II Floppy disks :-

- make sure of the system disk

التأكد من الأقراص  
 - disk drive A } ما ننتظر على A, B  
 - disk drive B } حتى نرى رسالة  
 الـ disk

### III Hard disk :-

Boot from disk A? Yes No

Try to enter C...? Yes No

Fixing the Booting area

problem on the booting area

- 1- Command.com or :-
- 2- system hidden files

HD is not found (invalid drive specification)

- Setup Problem or
- Hard disk problem

to identify one of them :-

- use diagnostic program (hardware test). Yes → Problem software

or

No → HD hardware problem  
 to check use → Disk partition table

this will be detected by the diagnostic test (mostly in the read/write flying heads).

Fig. 3.2 Handcrafted notes

6- Having completed the construction of the charts for a particular case, the information embedded in these charts is translated into rules using English-Like structures or Decision Tables. In this research, English-like structures is used to represent rules. These rules explain the relations between apparent syndromes and causes. These rules are further reviewed by the domain expert and any modifications are made. Table 3.1 shows some rules relating to solving the 'system does not boot problem', and the rest is shown in appendix B.

Following these steps, which are represented by the algorithm of Fig. 3.4, and during a two-month time span, four one-hour meetings were held per week with the domain experts (computer maintenance engineers). The process was terminated when a satisfactory amount of knowledge was acquired.

Table 3.1 Sample of rules about 'system does not boot' problem

- 
- R1 : If selftest is complete Then  
 - RAM is good and - ROM is good and - BIOS is good
- R2 : If selftest is not complete and there is no configuration changes Then problem is highly in the mother board and mostly in RAM .
- R3 : If system is not booting and there are error messages Then look at rules 4,5,6 .
- R4 : If bad memory indication Then problem is surely in RAM.
- R5 : If error code 1701 Then problem is surely in hard disk.
- R6 : If insert system disk and press any key Then problem is in operating system .
- R7 : If setup of the system is good and system still not booting Then problem is motherboard .

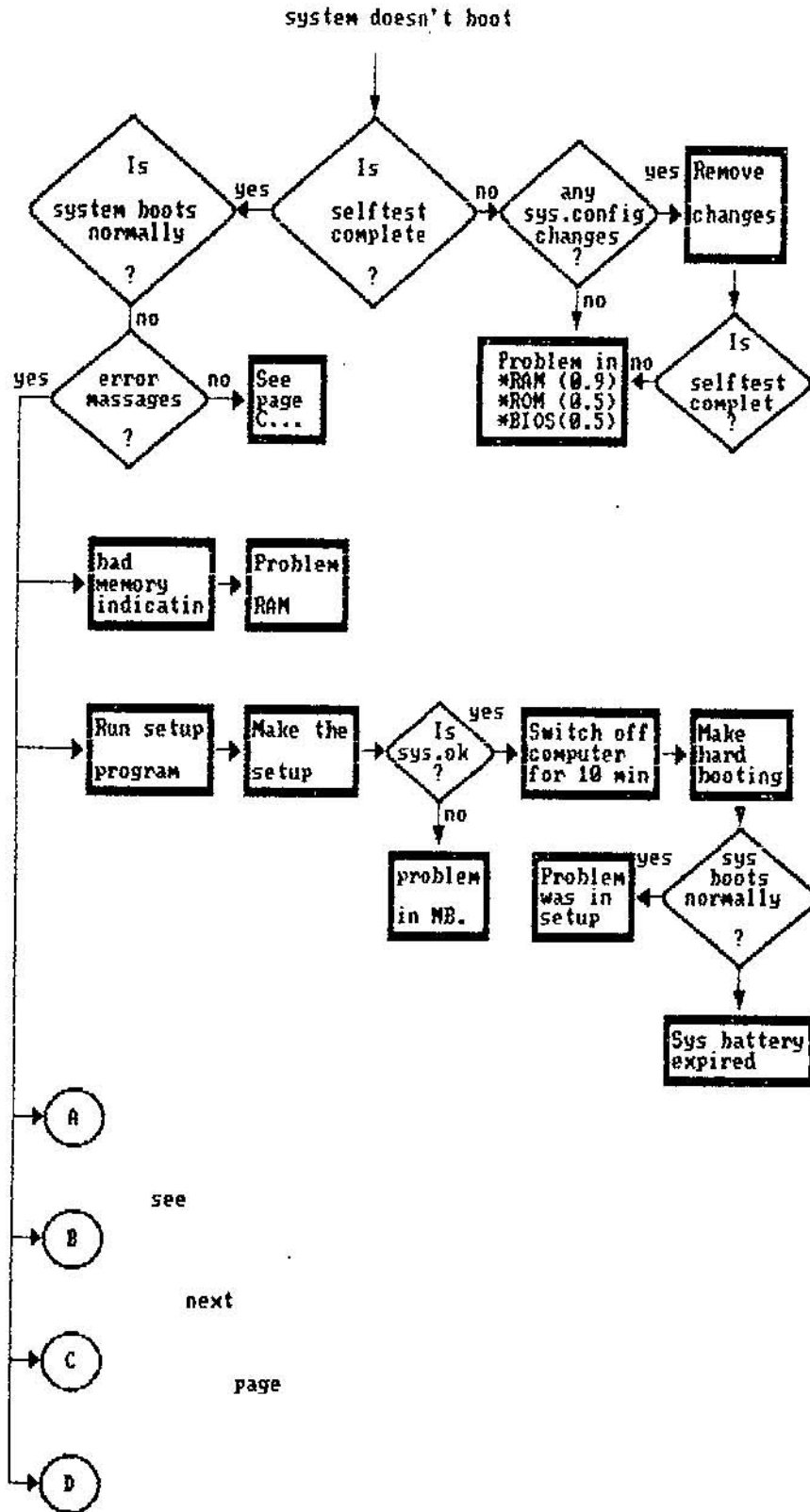


Fig. 3.3 A diagnostic flowchart

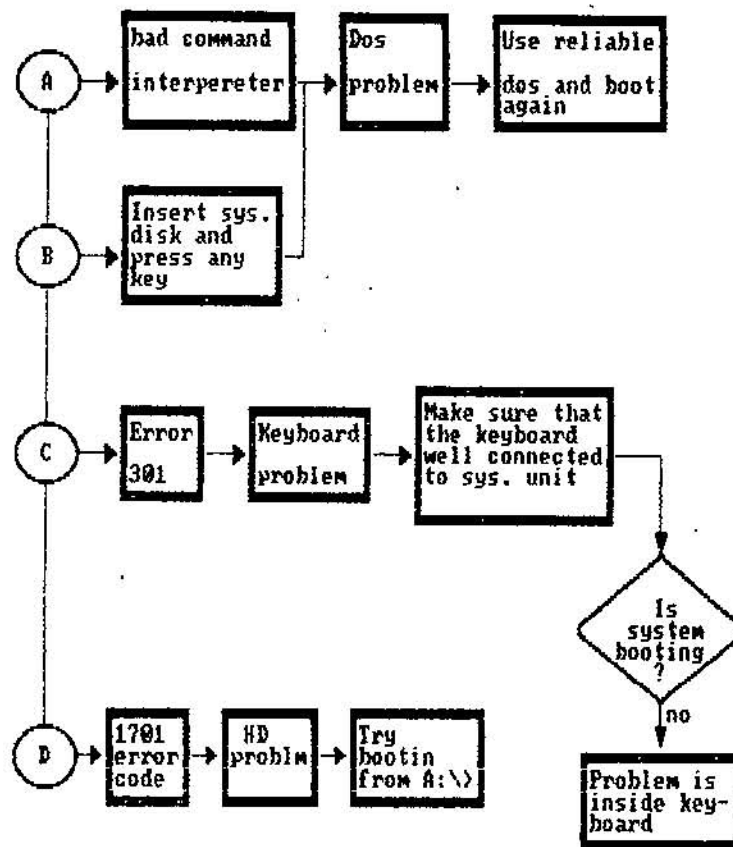


Fig. 3.3 Diagnostic flowchart .... continue...

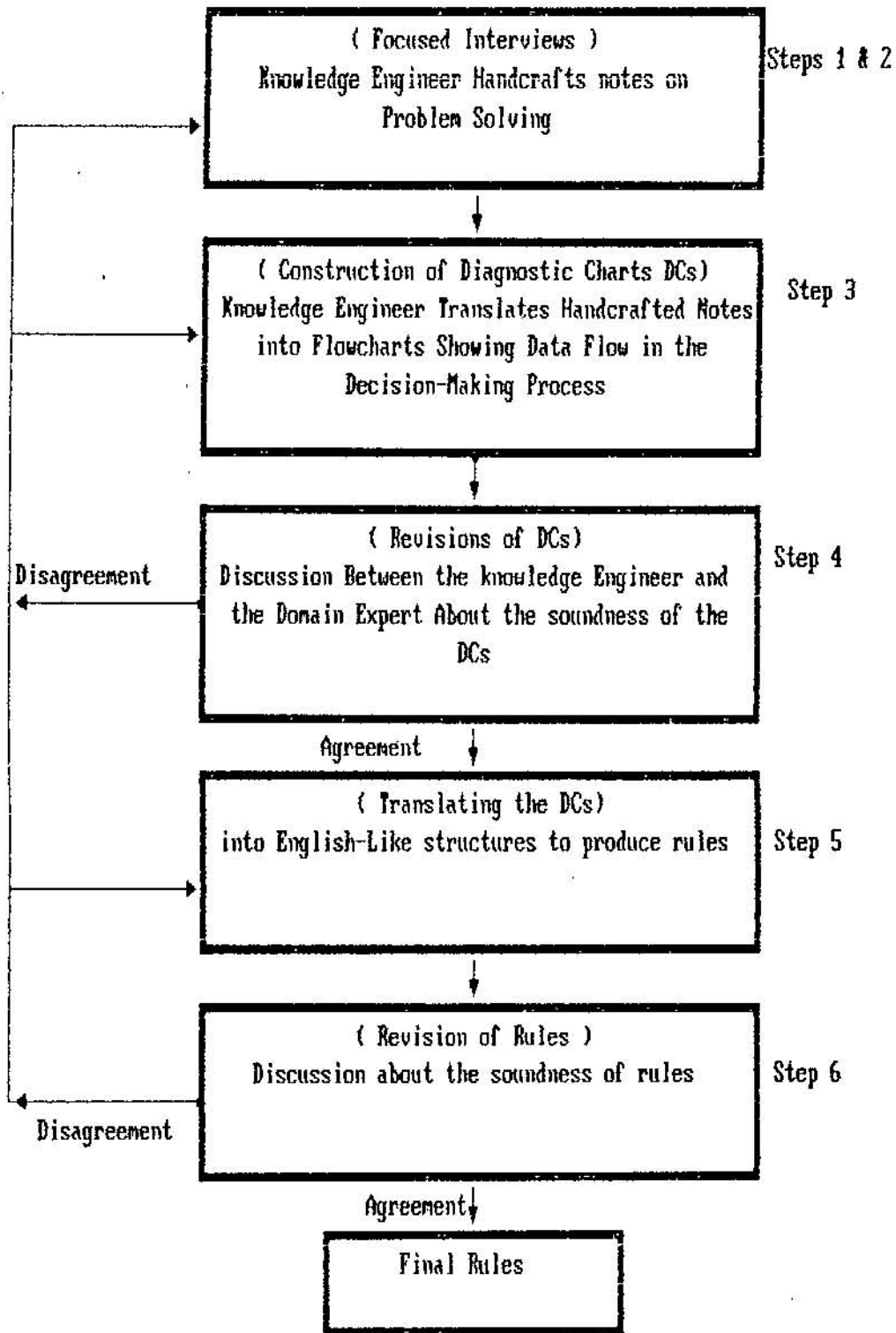


Fig 3.4 Flowchart of the interviewing algorithm

### 3.6 Formalization

The structured decision specifications from the previous step becomes an input to the formalization stage. The formalization of concepts is performed using a specific knowledge representation scheme. Knowledge that needs to be represented can take many forms. Such forms may include static knowledge (facts about objects), surface knowledge (superficial rules-of-thumb knowledge), deep knowledge (based on system design and functional requirements), procedural knowledge (step-by-step algorithms), and declarative knowledge (rules describing relationships) [5]. Actually, most knowledge types can be considered as procedural and declarative types. The effective representation of knowledge is a key issue in knowledge-based systems. Many knowledge representation schemes have been used where each scheme suits a particular type of knowledge. Three knowledge representation schemes are briefly explained here, namely : Frames, Semantic networks, and Production Rule systems.

#### 3.6.1 Frames

Frames were first suggested by Marvin Minsky in 1975. A frame is a data structure that provides a concise structural representations of useful relations. The frame concept consists of dividing knowledge into specific categories. They function much like tables that consist of a collection of slots, where each slot describes (instanciates) a certain attribute of the object being frame-represented. Frames stereotype (assemble) information making up usually a

complex entity through the suitable creation of slot that are used to store variables that characterize the frame. Frames generally have the following features :

- 1- It provides a structured representation of objects and classes of objects.
- 2- It provides a mechanism called inheritance that guides description movement from class description to individual description. This feature comes naturally as a result of the ability of frames to be linked through linking slots.
- 3- It allows one to determine description of objects in the absence of specific knowledge; this means that in the case where recent knowledge is not readily available about an object, the system can use already existing knowledge that is stored in that object-frame ( the default feature ).
- 4- It can represent not only objects being reasoned about, but also rules. When each rule is represented, rules can be grouped into classes, and the description of a rule can include arbitrary attributes of the rule. Fig. 3.5 shows a frame representing some airplane attributes.

Slot name	Slot Value
Engine Type	Volkswagen
Wing Span	75 m
Speed	700 Km/h
Weight	500 Tons

Fig. 3.5 Airplane frame with knowledge



### 3.6.2 Semantic Networks

A semantic network, in its simplest forms, is a graph whose nodes represent individual objects, and whose directed arcs represent binary relations. Semantic nets are ideal visualization tool that are widely used in AI to provide a mechanism to get started in the development of knowledge representation. Knowledge in semantic nets may be naturally organized to reflect hierarchies and to convey inheritance. Labeled arcs are used to denote relations between objects. Fig. 3.6 shows a semantic network representing a car.

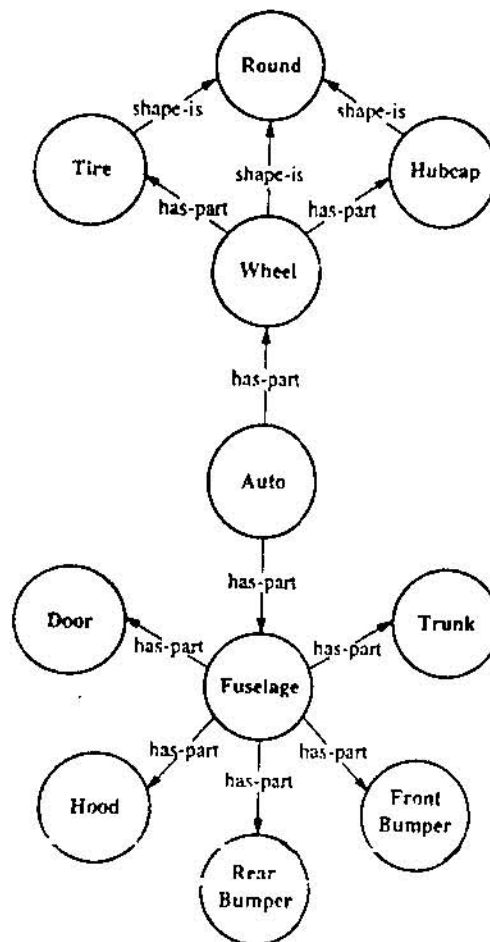


Fig. 3.6 Semantic net representation of a car [19]

### 3.6.3 Production Rules

Production rules are, in effect, a set of predicate calculus with an added prescriptive component indicating how the information in the rules can be used during reasoning. In predicate calculus, a predicate (relation) serves as a statement which defines objects and object relations. In computer terms, a predicate is a function that returns either a true or false value. Rule-based systems provide means to express situation-action heuristics in problem solving. Such systems are :

- 1- Based upon a set of IF-THEN implication based representation (rules) that modify the existing database.
- 2- Database of information.
- 3- Applied in narrowly defined problem domains, and are very suitable to convey surface knowledge accumulated by experience.

A production system contains a fact database, rule-base, and a set of meta rules that manipulate other rules. Production systems have many positive features such as the ease of modification, ease of exploring the current knowledge base, and the ease of following the reasoning process. They also have some disadvantages such as the inability to implement deep knowledge, and the lack of suitability for all applications such as the inability to represent static knowledge. For example, it is difficult to represent the dimensions of a box using rules. Also, it is impractical to represent basic principles such as physical laws using rules.

In practical knowledge-based systems, when the volume of the knowledge base get very large the search process for a particular solution takes longer time. This problem can be avoided by grouping the rules such that certain types of rules deal with an associated category of problems. In this fashion, the knowledge base is said to be structured [14].

#### 3.6.4 Knowledge Representation Scheme used in this Research

When choosing a knowledge representation scheme for a particular application, one should take care of the type of knowledge involved. It is clear from the previous discussion in this chapter that the knowledge involved in this research is heavily empirical, and best described as rules-of-thumb or heuristic knowledge. Factual knowledge is also present in this project to define static and dynamic object attributes. Heuristic knowledge is best represented by production rules and factual knowledge by frames. Thus, these two complementary techniques are used in this project. The use of these two methods facilitates the process of building an inference engine, as will be illustrated in the implementation phase. The inference engine consists primarily of meta-rules that guide the search process, where the results of this process are stored in frames for later use.

The search process, or control strategy, is discussed in the implementation phase. This process can follow forward chaining and/or backward chaining. The certainty factor is not discussed here assuming that the all the acquired knowledge is 100% certain.

### 3.7 Implementation of the System

During the implementation stage the expert system comes to be a reality, when the concepts are transformed into a working computer program that is usable and testable. During this phase a suitable programming language is chosen for the development of the expert system. The knowledge engineer starts by developing a prototype system that is used as a means for further modifications and enhancements. This development process is iterative and continuous, and never ends as long as new ideas and system features are proposed by the designers. In this research, a pilot system has been developed that needs some more enhancements to be used as a practical system. The block diagram of the expert system is shown in Fig.1.2. It consists of the user-interface, knowledge base, inference engine, and reasoning and learning facilities. Modular programming strategy was followed during implementation. The following sections explain the steps of the implementation phase.

#### 3.7.1 Implementing Language

Expert systems can be developed using any programming language. However, special programming languages have been developed for coding expert systems. These languages such as Lisp and Prolog supports the symbolic processing and non-procedural solution methods. For the rapid production of expert systems, new tools called expert system shells have also been developed. These shells are expert systems that have fixed system structure, and lack the presence of the

domain knowledge. The system in this project is implemented using dBase IV version 1.1. Although this programming language is highly procedural, it supports a variety of desirable AI programming features such as modular programming, separation of the knowledge from the control structure, string-pattern matching, capability of representing symbolic data, and the interactive system communication capability. This language is available, easy to use, provides English-Like syntax, and flexible. The system is developed to run on IBM PCs and compatibles, under the MS-DOS operating system; in a single-user mode. A separate manual of the system is attached for further reference. The major part of the computer program list (source code) is shown in appendix C.

### 3.7.2 Knowledge Base Structure

The knowledge base contains the actual knowledge that can be used in problem solving. This knowledge is stored according to a defined structure that holds the knowledge items. So the knowledge base constitutes a suitable knowledge structure plus the knowledge itself. As said before, Frames are chosen to represent facts about objects, and rules. Frames can easily be constructed using dBase IV. Each frame comprises a set of related data items that can uniquely identify a particular object through its values stored in the frame slots. Fig. 3.7 shows the different frames used in this project, and also the data slots (fields) associated with each frame. These frames are described below :

-- rule-base frame : this frame provides data items that apply

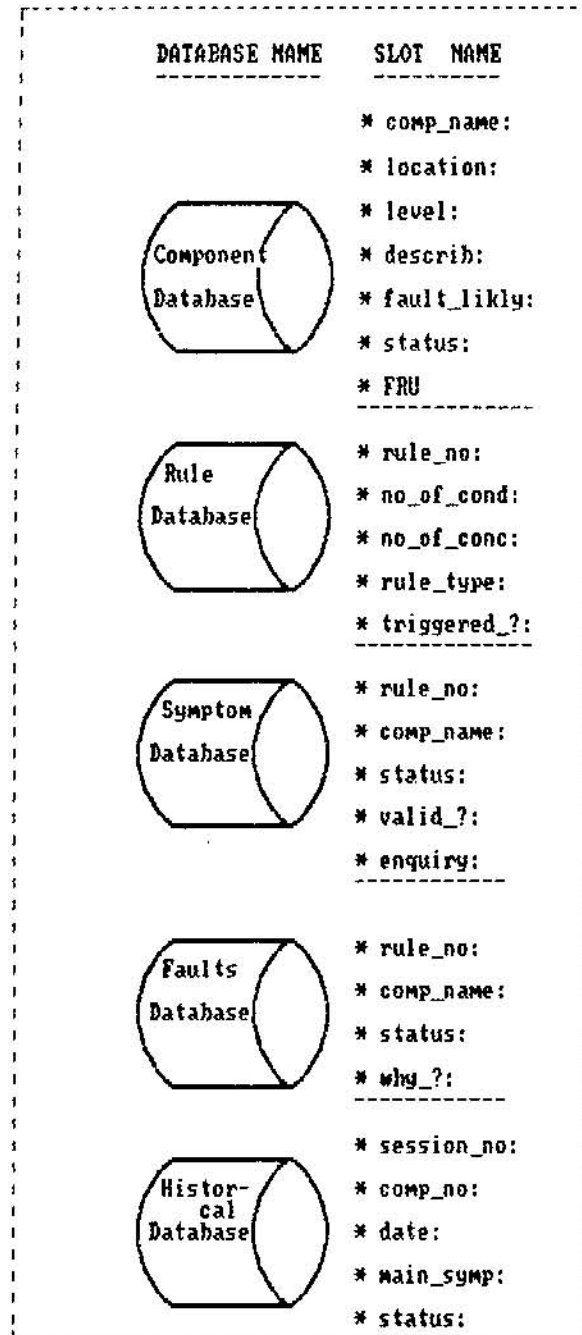


Fig. 3.7 Database structure

to rules such as the rule number, rule type, number of conditions, number of conclusions, and the status of the rule.

- symptom (condition) frame : it provides data items to describe a certain symptom such as symptom name, rule number relating to this symptom, and another frame slot to check if symptom is there.
- faults ( conclusion) frame : it provides data slots that can recognize a certain fault in the system. Data slots include conclusion name, rule number, and a status-checking slot.
- component frame : it provides fields about component attributes which can link all components together to make up the whole system. A slot is used to tag faulty components.
- decision frame : this frame is used as temporary storage of decisions made by the system. It is a part of the working memory and supports fields such as the decision name, decision number, faulty components ..etc.
- data frame : this frame has the same data structure as that of the decision, but data is stored here permanently for later reference. The knowledge base also contains knowledge item that are not represented by frames. These are actually the production rules that guide the process of selecting certain knowledge items stored in the frames. These rules are shown in appendix C as part of the program source code.

### 3.7.3 Inferencing Process

The inference process constitutes the total activities that the expert system performs to reach a decision regarding a certain problem. This inference process include all the operations that the system does starting from data collection, making decisions, and finally explaining these decisions. This process is controlled by a module called the inference engine. The inference engine is a group of computer programs that control the whole system processes. To better understand the system operation, a diagnostic session is presented in the next chapter, with an explanation describing how the system reached the decision. Also, it is beneficial to look at the inference engine structure. The inference engine developed in this research is composed of the following submodules :

-- Data collector : collects data from the databases, and from the user, and stores this data in the working memory for later use. Data is taken from the user through Y/N questions, and multiple answer questions. This part actually could be considered as a part of the user-interface.

-- Scheduler : determines which facts, rules, and relationships should be used by selecting a certain control method. Forward chaining method is used in this project. Following this strategy, the system begins with an initial data and proceeds towards the goal state by matching the condition parts of the rules with given data, trying to figure out new data contained in the action parts of the rules. This



method is used in the developed system because the number of the initial state is less than the number of the final (goal) states. On the other hand, backward chaining strategy starts from the goal state by proposing a hypothesis regarding solving a problem (this hypothesis is usually embedded in the rules consequents), and then tries to prove this hypothesis by proving the conditions of these rules which in turn can be considered as goals. When the database gets larger, a mix of both strategies can be used to increase program efficiency.

-- Interpreter : processes the data that is collected through the data collector, by matching up rules with known facts and performing actions specified by these rules such as changing the status of faulty components.

-- Scanner : this module scans the component database to check the component status, figure out the faulty components, and displays the decision. The inference engine structure is shown in Fig. 3.8.

#### 3.7.4 Explanation and Learning

The expert system that has been developed in this project is a rule-based system that relies heavily on experience where problem patterns are associated with straight forward solutions. Thus, the explanation (reasoning) given by the system is based on tracking the rules that are fired during decision-making processes. Each rule has a 'WHY'-slot that reason the actions specified by the rule, and this knowledge slot is inputted as part of the expert-defined information. As part of the explanation module, the user can

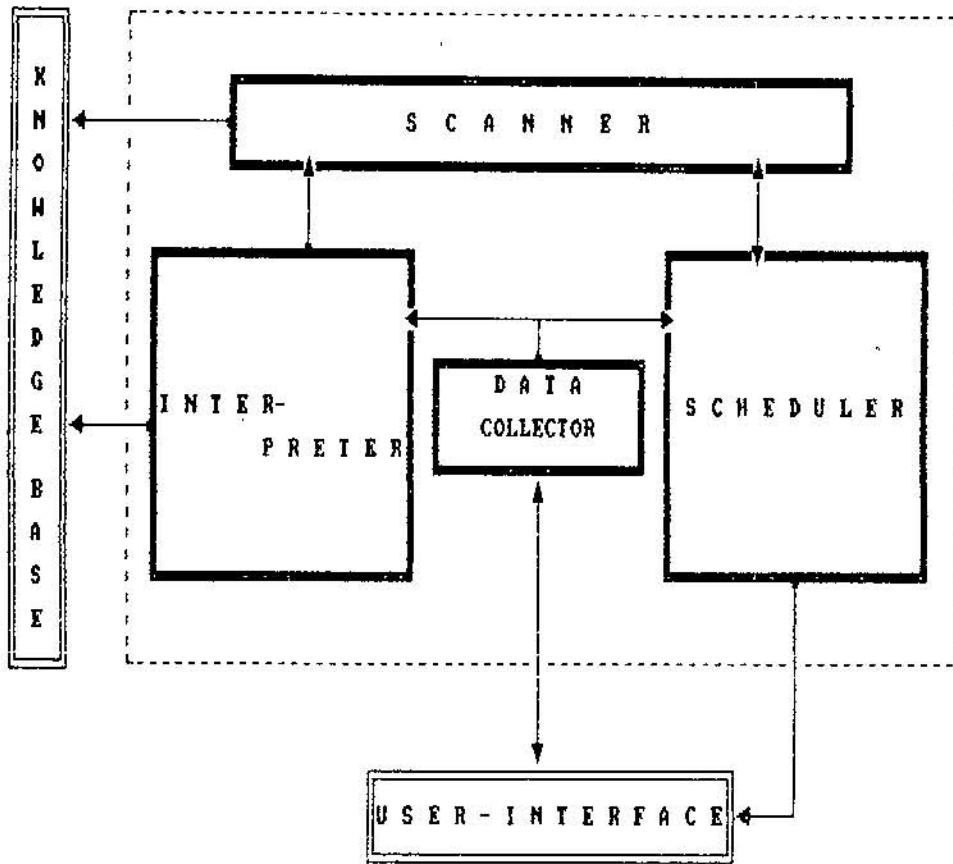


Fig. 3.8 Inference engine structure

## CHAPTER FOUR

### A CASE STUDY

#### 4.1 General

In this chapter, a diagnostic session is performed by the developed system. This session clearly shows the way the system communicates with users, and the different activities that users can do, (see Fig.4.1). These activity options are displayed to the user through multilayer window menus. Also, a lot of guiding messages are present to direct the operator to the correct action that avoids the possibility of making wrong logical decisions because of previous variables values.

As stated in the objectives of this research, the system offers two types of consultations. Preliminary consultation gives brief idea about the main system module that could suffer the malfunction. This diagnostic mode is suitable for ordinary people who do not have the computer maintenance knowledge. No tests or measurements are required to be performed at this stage. Deep diagnosis goes further beyond the former one where it is necessary sometimes to perform tests and measurements on the computer system circuit boards. These tests require the system case to be removed, careful use of the measuring devices, and correct interpretation of the readings taken. These activities are best performed by a technician who is familiar with them.

At the end of each consultation, the system gives an advice (recommendation) about the most probable faulty component that caused the malfunction. Also, the system can

show its own reasons for making such a recommendation. This explanation is based on experience and judgment obtained from the domain experts. The explanations are shallow because they are not based on the basic principles of the system design, and the system is rule-base.

Next is shown the computer printouts for the performed diagnostic sessions, together with the system menus. Two sessions are shown : one Preliminary and the other is Deep. After that an explanation about the way system made the decision is given in terms of the general sequence of operations that were performed to make the decision. Then a real human expert-customer diagnostic session is made to make a comparison between the expert system and the human expert.

WELCOME TO AL-GHANIM'S EXPERT SYSTEM

---

EXPERT EXPERT EXPERT EXPERT EXPERT EXPERT  
 \*\*\*\*\*

M.Sc. Thesis Project  
 University of Jordan  
 Faculty of Graduate Studies  
 Industrial Engineering Department

=====  
 C Copyrights reserved Amjed Al-Ghanim , 1991  
 =====

---

Notes :-

- \* The system interface is designed for your convenience .
- \* Just follow what the system tells you to do .
- \* Before you proceed to either type of diagnosis ,  
 please, clear the working memory, next menu .

-----  
 Press Any Key To Reach The Main Menu, Q - to quit the system  
 -----

WELCOME TO AL-GHANIM'S EXPERT SYSTEM

Here is the Main Menu	
Task Type	Task Code
-----	-----
Edit Knowledge Base	E
Run a Consultation	R
Clear working memory	C
Exit the System	Q
=====	

Please enter  
Code

Fig. 4.1 System heading & main menu

### 4.2 Preliminary Diagnosis Session

Before you start a diagnostic session, be sure that the checks made on your PC system :

- ```

=====
* external power outlet is hot
* system power cable connection at both sides
* monitor power cable connection at both sides
* logic cables pin orientation at both sides
* printer power and logic cables connections
* DOS version is correct and reliable
=====

```

```

Diagnosis will start now, answer the system's questions
according to the described format
You can perform two types of consultations :
Preliminary Diagnosis [ P ] -- Deep Diagnosis [ D ]
Quit to previous menu [ Q ]
Enter your choice ....

```

#### Preliminary Diagnosis Session

Preliminary diagnosis aims at identifying the main system module which suffers the malfunction . Generally no exact specification of the problem diagnosis is given at this stage . Now, SWITCH ON your PC system and press any key to start the session

```

Question #          1
---Do you hear system fan operating :
A- normally (contineously) ?
B- operate at start then stops ( unusual operation) ?
   Chose A or B

```

```

Question #          2
---Is monitor power LED lighting ? [ Y/N ] n
Question #          3
--- If you replace the monitor power cable with another one
   or replace the electric fuses
   (use the system power cable), does it operate normally ?

```

Fig. 4.2 Preliminary diagnosis session

```

--- If you replace the monitor power cable with another one
    or replace the electric fuses
    (use the system power cable), does it operate normally ? y
Question #          4
---Do you have a printer connected to your system and
powered ?

```

```

Question #          5
---Is printer 'on line' LED on, is it in ready mode ? n

----- End of preliminary questions -----
If you want the questions to be repeated for you, Press Y now
    Otherwise, Press any key to continue

```

Press any key to see the diagnostic decision  
that the system has made....

```

If you want the questions to be repeated for you, Press Y now
    Otherwise, Press any key to continue

```

```

Decision Number          5
*****
The following components are likely to be faulty :
-----
Component name           Field replacable unit
=====
Monitor Power Cable      monitor power cable
Printer                   printer

    Press ( Q ) to quit the system.....
    Press any key to the enter the explanation facility ....

```

Fig. 4.2 Continue

---

DECISION --- EXPLANATION FACILITY

---

Decision about a previous session -- 2 \*

Expalaning the decision of the present session -- 3

Explaining the decision of a pevious session -- 4 \*

---- option labeled with \* are not implemented ----  
 Please enter your choice number .....

---

Because this Expert System is Rule-Based, the explanation given here is not deep . Explanation is based on what the rules say about a certain problem .

-----

---

Decision #                    5  
 =====  
 -----> Monitor Power Cable  
 -----> Printer

Explanation

\*\*\*\*\*

After replacing monitor power cable, monitor works normally implying that problem is in monitor power cable  
 This decision is very rough, the error may be in the main system unit but mostly in the printer  
 Press any key to continue...

---

Fig. 4.2 Continue



### 4.3 Deep Diagnosis Session

#### Deep diagnosis session -- Part A

---

Deep diagnosis gives you a better idea about system malfunctions through asking more questions, and may be by carrying out certain tests that the system asks you to do. This deep diagnosis process is better performed if the user (person answering the questions) is a computer technician who is capable of making some tests.

-- Any key to continue diagnosis ---  
 -- Back to previous menu R --

Please enter your choice to continue

---

#### Electricity Check

The following questions identify a power problem. If power is present, then the main faulty module(s) is specified. Then the system asks you several questions about that module. Press any key

---

Question # 1  
 A- normally (continuously) ? A  
 B- operate at start then stops (unusual operation) ? B  
 C- does not operate at all ? C  
 Chose A or B or C

---

Chose A or B or C a

Question # 2  
 Is system able to boot successfully :  
 Look at the selftest on screen, and observe the drives

Question # 3  
 If your system has a power indicating LED :  
 Is it lighting continuously ? [Y/N] y

Question # 4  
 Is monitor showing normal screen response ? [ Y/N ]

---

Fig. 4.3 Deep diagnosis session

---

Your system has faults in one of the main modules :  
 Main modules are : Main system unit  
                           Monitor unit  
                           Printer unit

So press any key to continue .... to part B

---

Deep diagnosis session --- B

---

Advanced Question #           1  
 Is system selftest completed successfully [Y/N] ?

---

Deep diagnosis session --- B

---

Advanced Question #           2  
 Is there any error messages (after or during selftest) ? n  
 Advanced Question #           3  
 Do you boot from :  
 A-- hard disk & a floppy disk  
 B-- Tow Floppy disks  
 Enter you answer A or B

---

Deep diagnosis session --- B

---

B-- Tow Floppy disks  
 Enter you answer A or B b  
 Advanced Question #           4  
 Now you have tow floppies, and you are booting from drive A ,  
 Now, try to boot from drive B, does it boot ? y  
 Advanced Question #           5  
 Do you have a new system battery ?

---

Fig. 4.3 Continue

Deep diagnosis session --- B

---

Now you have tow floppies, and you are booting from drive A  
 Now, try to boot from drive B, does it boot ? y  
 Advanced Question # 5  
 Do you have a new system battery ? y  
 Advanced Question # 6  
 Do you have : XT-system (A) or AT-system (B)  
 Enter your choice ...? a

---

Deep Diagnosis Session 10

=====

| These components are likely to be faulty | Feild replacable u |
|------------------------------------------|--------------------|
| -----                                    |                    |
| --> Main System Unit                     | main system        |
| --> Ram, System Main Memory              | Mother Boar        |

Do you want to see the explanation ?[Y/N]..

\*\*\*\*\* Explanation \*\*\*\*\*

Since the system isn't booting successfully, problem is  
 in the main system unit:RAM,ROM,drives,cards,or software  
 When the selftest is not complete, the RAM is mostly the  
 component . Problem is surely in motherboard .  
 Press any key to continue...

---

Fig. 4.3 Continue

#### 4.4 Decision-Making Process

As seen from the computer runs, during a diagnostic session the system asks the user a number of questions after which the system makes the recommendation. The sequence of questions is not the same with every diagnostic session. What questions the system asks during any given session depends on the answers the user give in response to these questions.

The processes followed by the system during diagnosis are commensurate with the inference engine structure since the inference engine is the controller of whole system. The general (general because processes are sometimes overlapping) sequence of inferencing processes goes through the following phases :

Information collection phase : during this phase, and through asking questions, the system collects general information about the system conditions (symptoms) during preliminary diagnosis.

These preliminary questions are shown in Fig.4.2. In the case of deep diagnosis, the system proceeds further to collect detailed data about the system conditions by asking the user to do some tests and measurements. This is shown in Fig.4.3, deep diagnosis part B. In both types of sessions, the system stores the gathered information in a working memory, and also invokes the symptoms data base.

Interpretation phase : during this phase, the system interprets the collected information with respect to the knowledge base, i.e. the system gives meanings to acquired

data. Based on the interpretation the system does for a particular problem, the interpreter modifies the component data base in order to tag the faulty components.

Scanning phase : during this phase, the scanner scans the component data base, and selects those faulty-tagged ones. Those components as well as other related information (such as the reasons for tagging a component as faulty) are then stored in a temporary output file (Decision file). After the decision is displayed, this information is stored in the historical database.

Scheduling phase : actually this phase is not clearly defined in time, because the scheduler operates interchangeably during the previous phases. The scheduler determines when to ask a particular question (collection phase), takes information from the interpreter about the status of component, and also determines the decisions problem at hand. All these phases are shown in Fig. 4.4.

During the inferencing process, forward chaining was used where new information are obtained about the system status as a result of invoking the rules conditions and evaluating them against the input data. A goal is reached when no more input information becomes useful. Also, a goal is reached when a component (components) that belongs to the knowledge base is found to be responsible for the system state.

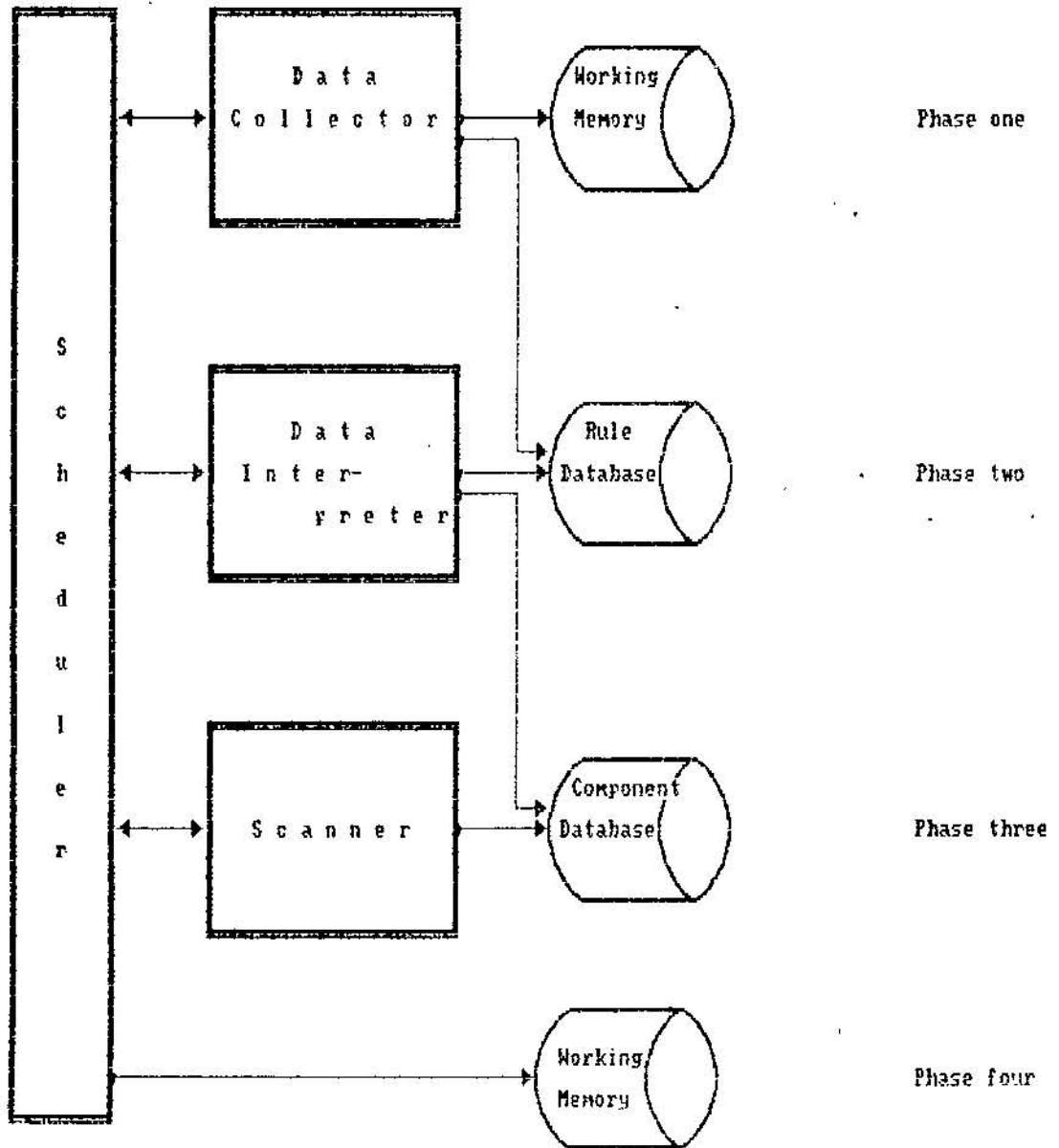


Fig. 4.4 Inference process

#### 4.5 A Human Expert-Customer Diagnostic Dialogue

A diagnostic session was held with the human expert whose knowledge was captured in this research. The session ran in the form of a dialog between the expert and a customer who has a non-working computer system. The dialog ran as follows :

CUSTOMER : My computer is not working ? It does not boot.  
 EXPERT : Are you sure that your system is well installed and powered ?  
 CUSTOMER : Yes.  
 EXPERT : OK, when you switch on your computer, You see the memory count. This is the selftest of the RAM, is it complete ?  
 CUSTOMER : The system counts until the number 256 appears on screen, and then stops.  
 EXPERT : After that happens, is there any error messages ?  
 CUSTOMER : YES.  
 EXPERT : What is the error message ?  
 CUSTOMER : It is 'Bad memory '  
 EXPERT : OK, your problem is most likely in the RAM part of the motherboard.  
 CUSTOMER : What is the solution now ?  
 EXPERT : I'll bring you a motherboard and check upon the RAM.  
 CUSTOMER : Thank you.

It seen from the above dialogue that problem turned out to be in the motherboard, and mostly in the RAM part. As compared with the human expert-customer dialogue, the expert system asks more questions about the faulty system conditions, and the system gives first a rough answer when it recommends that the problem is in the main system unit. Then the system asks more questions, after which it could determine that the problem is in the RAM, the same decision of the human expert. These results are obvious from the deep diagnostic session.

## CHAPTER FIVE

### CONCLUSIONS AND RECOMMENDATIONS

It is obvious that the development process of the expert system is a two-fold job. The engineering part concerns the use of the system in engineering applications, the knowledge acquisition and representation, and the decision-making process through the appropriate design of the databases. The computer science part concerns the transformation of these concepts into a working computer program. This part mainly includes the system coding and partial database design.

The final output of this research is a working expert system software that could be useful in computer maintenance organizations. The different data bases were designed to minimize the required storage, and to best represent the knowledge items involved. The inference engine structure runs parallel to the designed phases of the inference process; namely the data collector, interpreter, scanner, and scheduler. The main concepts underlying expert systems, such as knowledge acquisition, knowledge representation, and reasoning, were all implemented in the final system.

The developed expert system is to be used in a maintenance organization as an engineering tool. The system offers both technical and managerial assistance to the maintenance organization. It aids the diagnosticians in solving diagnostic problems. It also facilitates operational activities as it can replace (when it has enough practical



knowledge) the human expert, gives faster response, and produces better quality work and documentations. Thus, this expert system can enhance the company operation by supporting the already existing information systems.

The expert system developed in this project needs more knowledge to be used as a practical system. This knowledge-addition process is continuous, and requires more domain experts and time than those that were available for this research. But the system can be updated to make it practical.

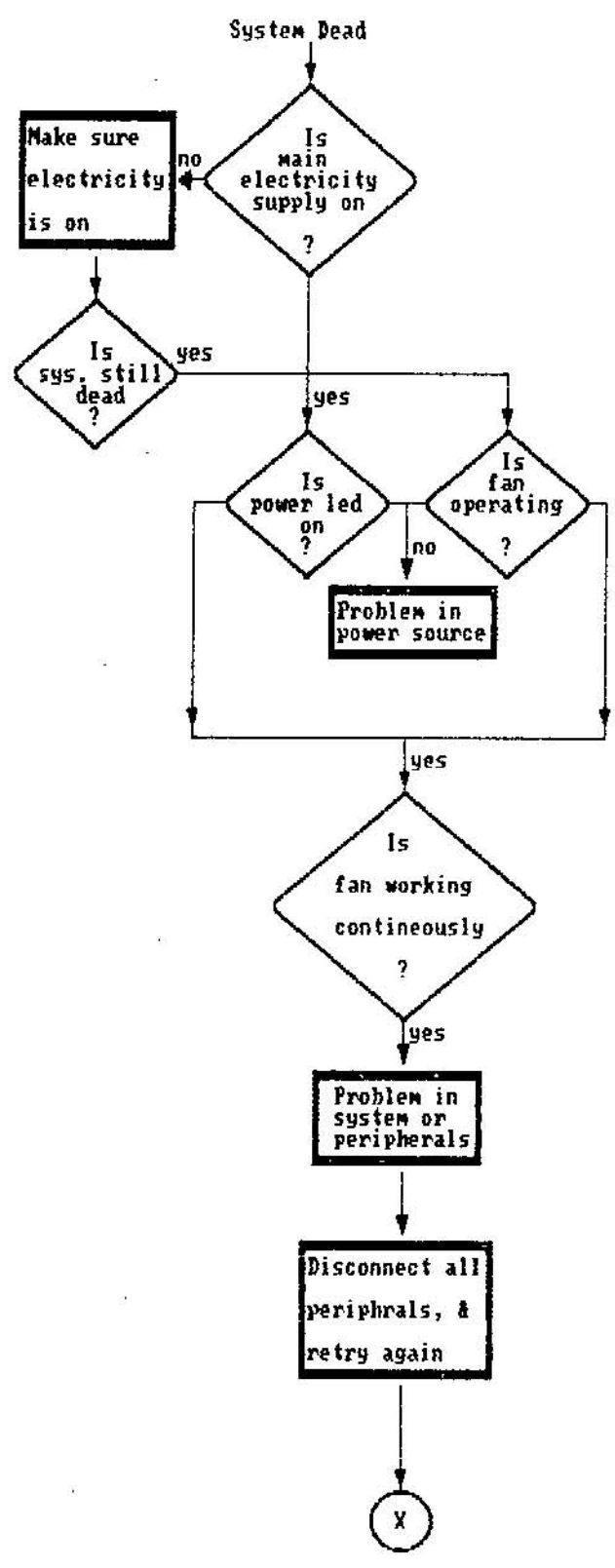
The following recommendations are made :

1- Since the developed system is a diagnostic system whose output is a set of faults that a certain machine could suffer, then this system can be integrated with a maintenance management system that has system faults data as part of its inputs. The integrated system has then faulty machine symptoms as part of its input data. So it is highly recommended that this system be adopted by computer maintenance organizations, and change menus to work in the Arabic mode.

2- For future research, the following are suggested :

a- choosing different knowledge representation schemes and comparing among them to decide upon the best knowledge representation mixture.

b- choosing different control strategies and making the same comparison at different knowledge base volumes, and deciding on the optimal control strategy for a given knowledge base.



Next page

Fig. A.1 Diagnostic flowchart : system dead

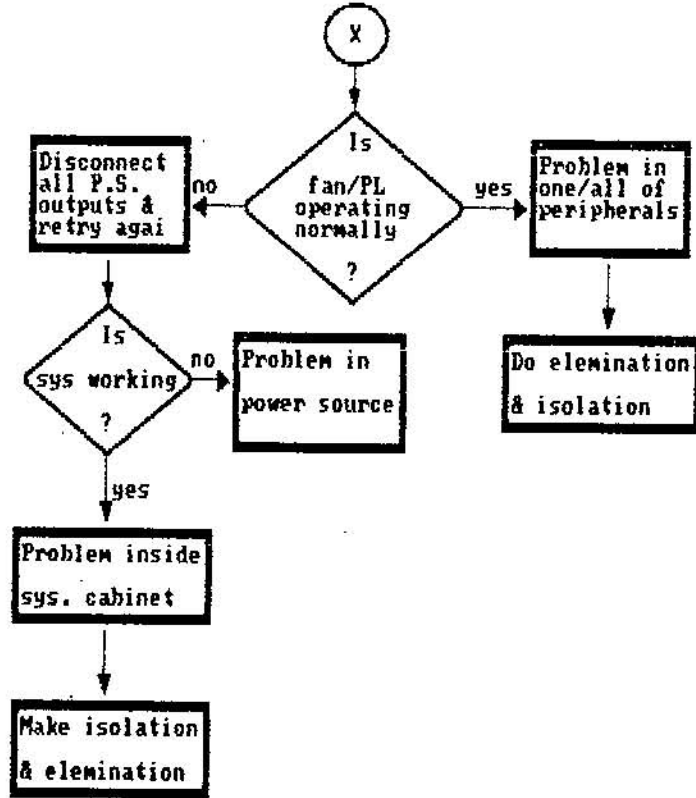


Fig.A4 continue...

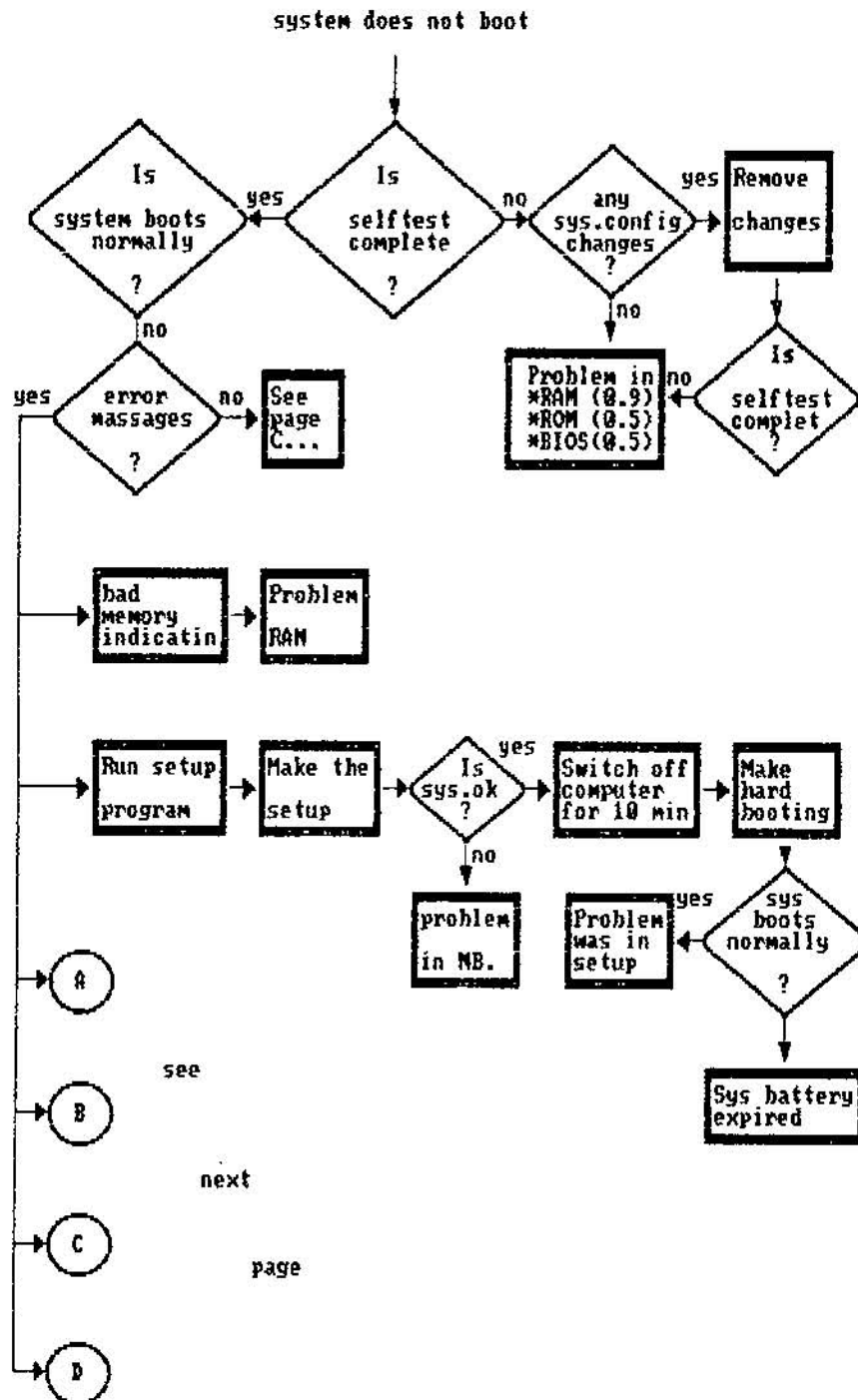


Fig.A.2 Diagnostic flowchart : system does not boot

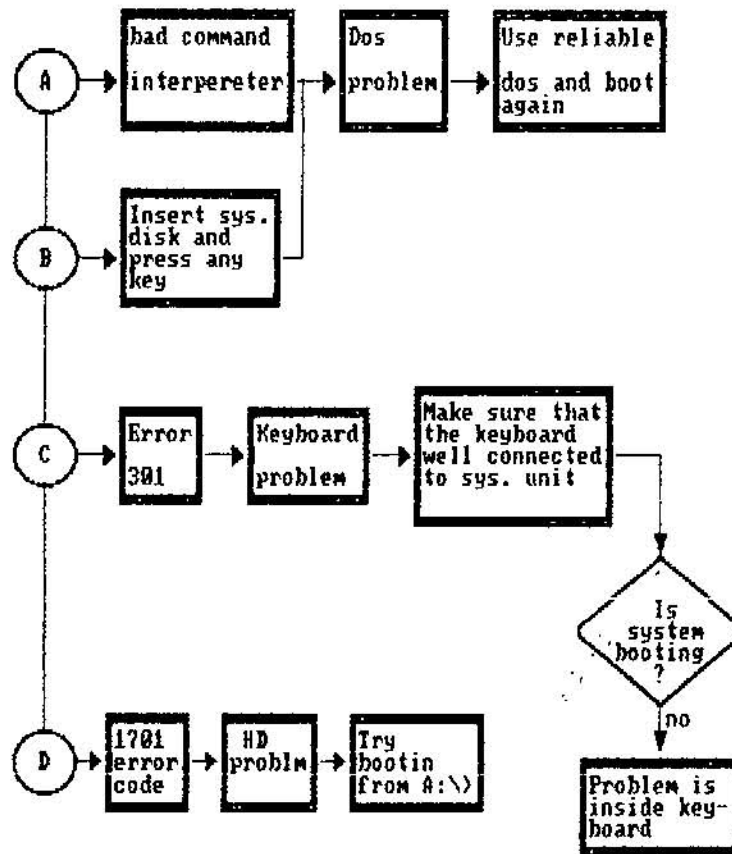


Fig.A.2 continue...

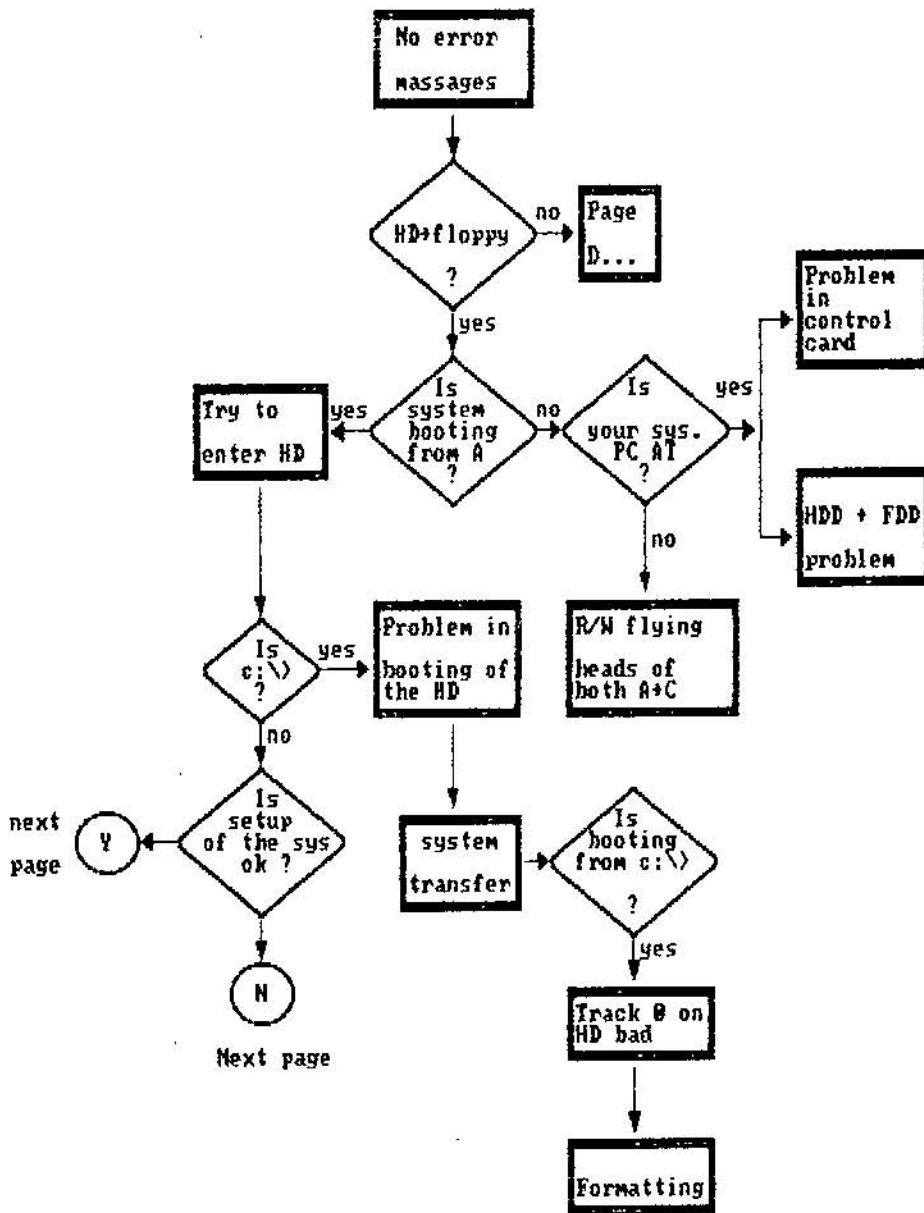


Fig.A.2 continue...

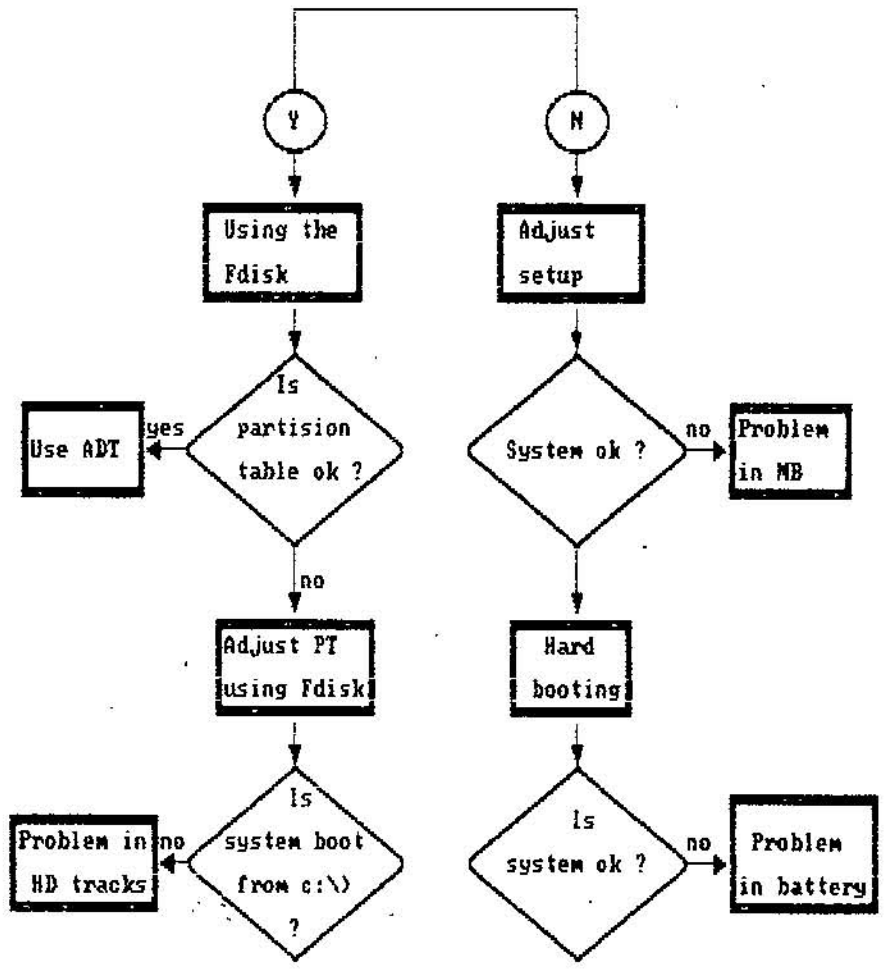


Fig.A.2 continue...

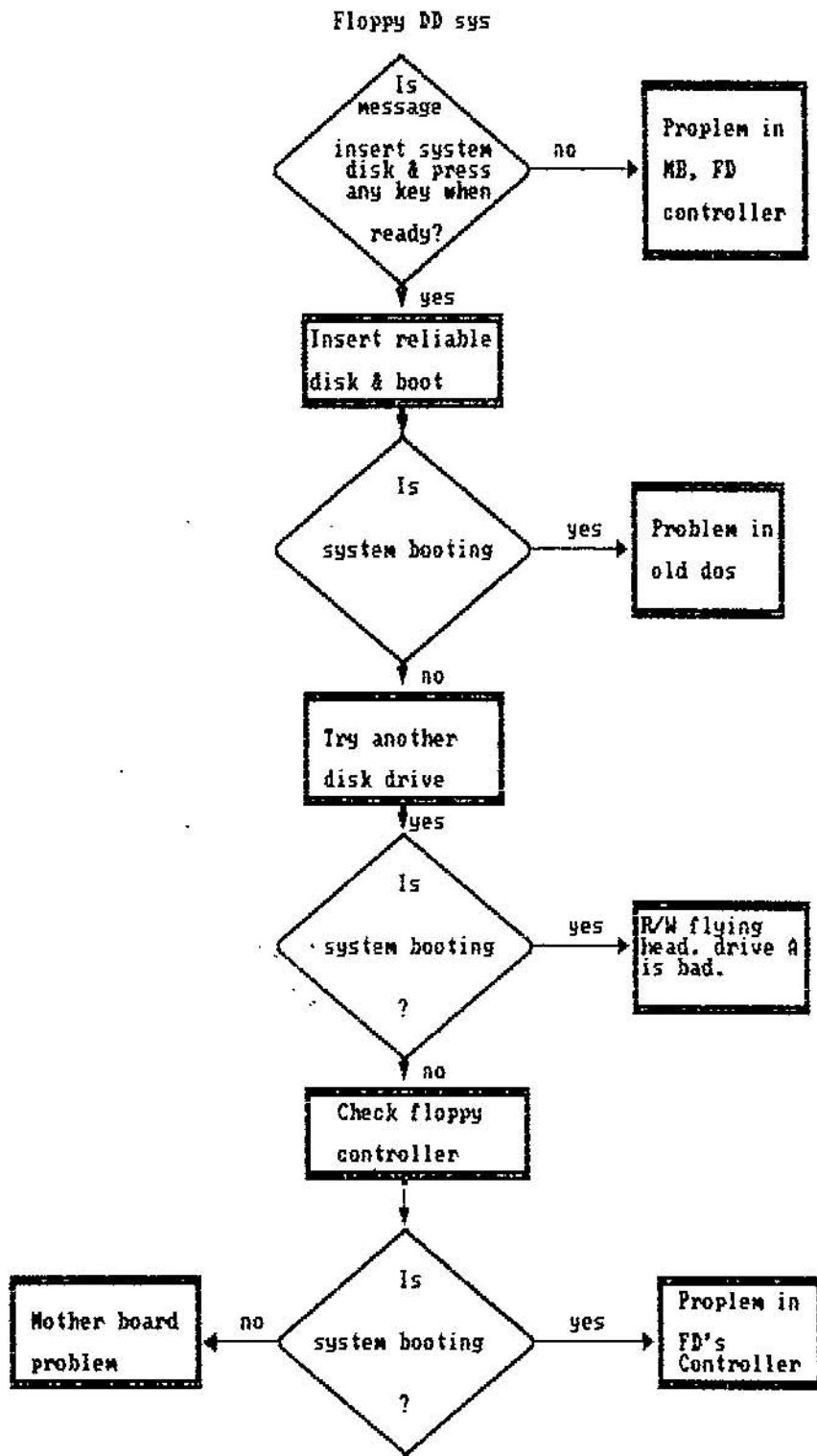


Fig.A.2 continue...



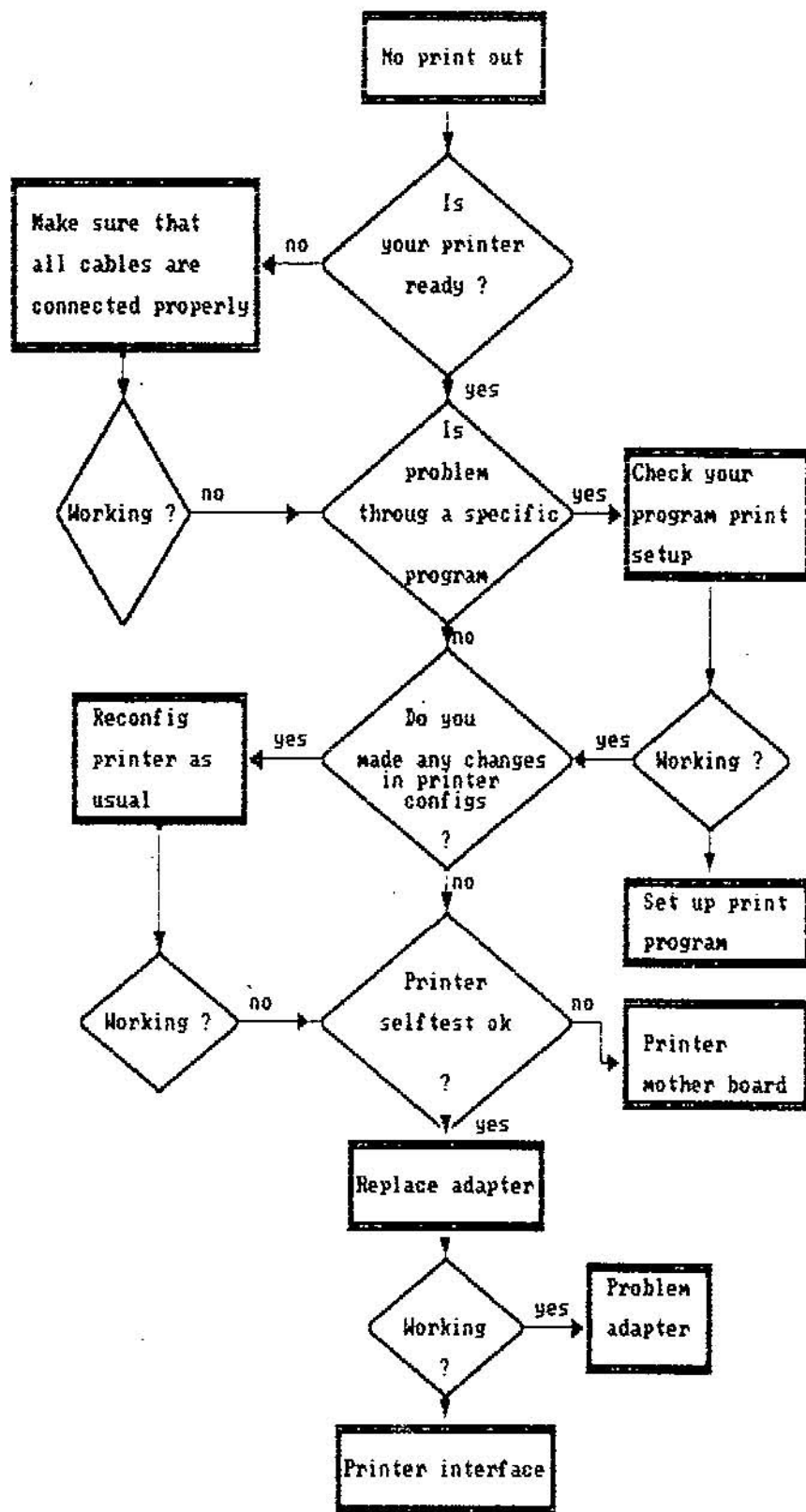


Fig.A.3 Diagnostic flowchrt : no printout

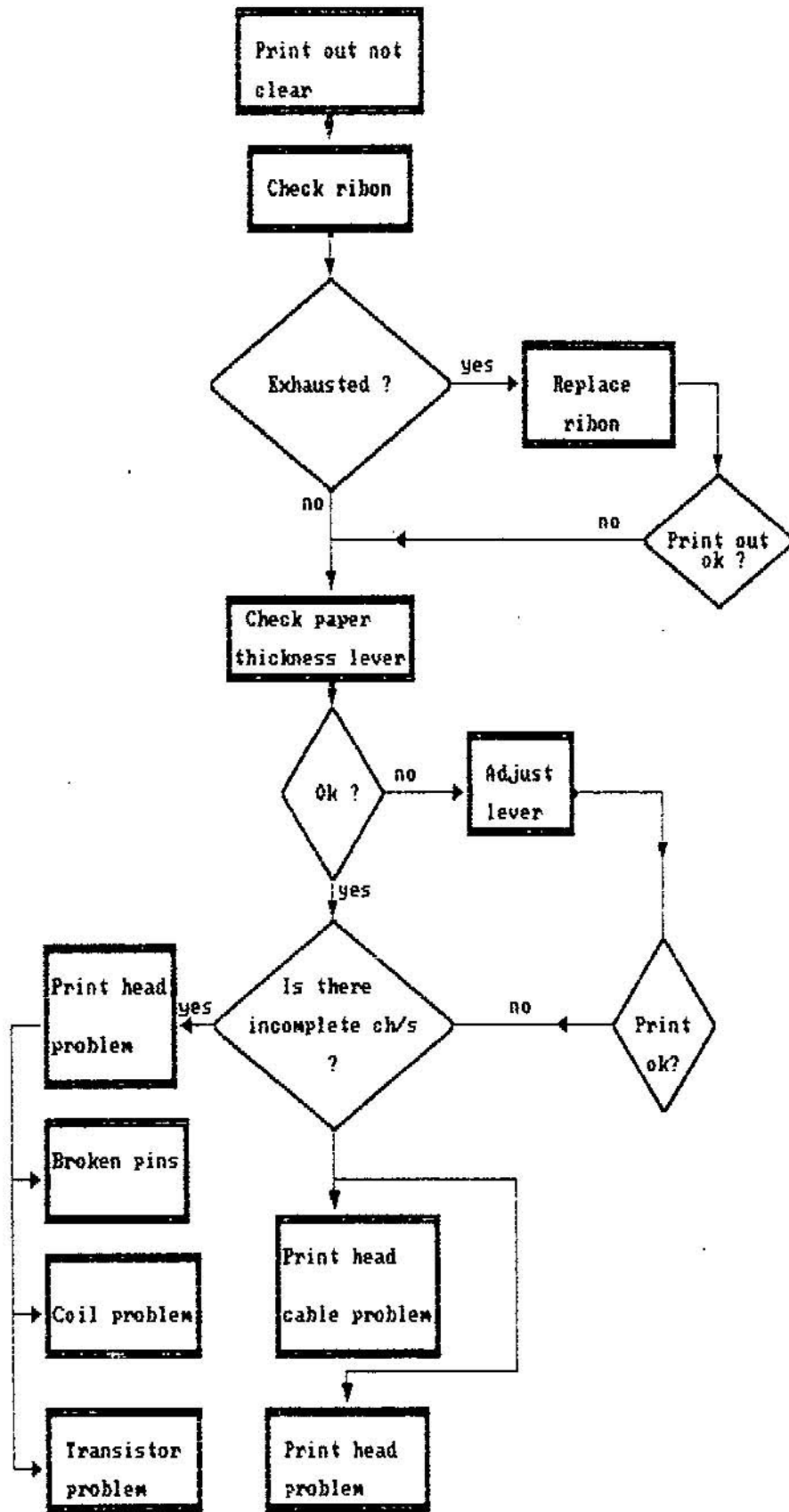


Fig.A.4 Diagnostic flowchart : printout not clear

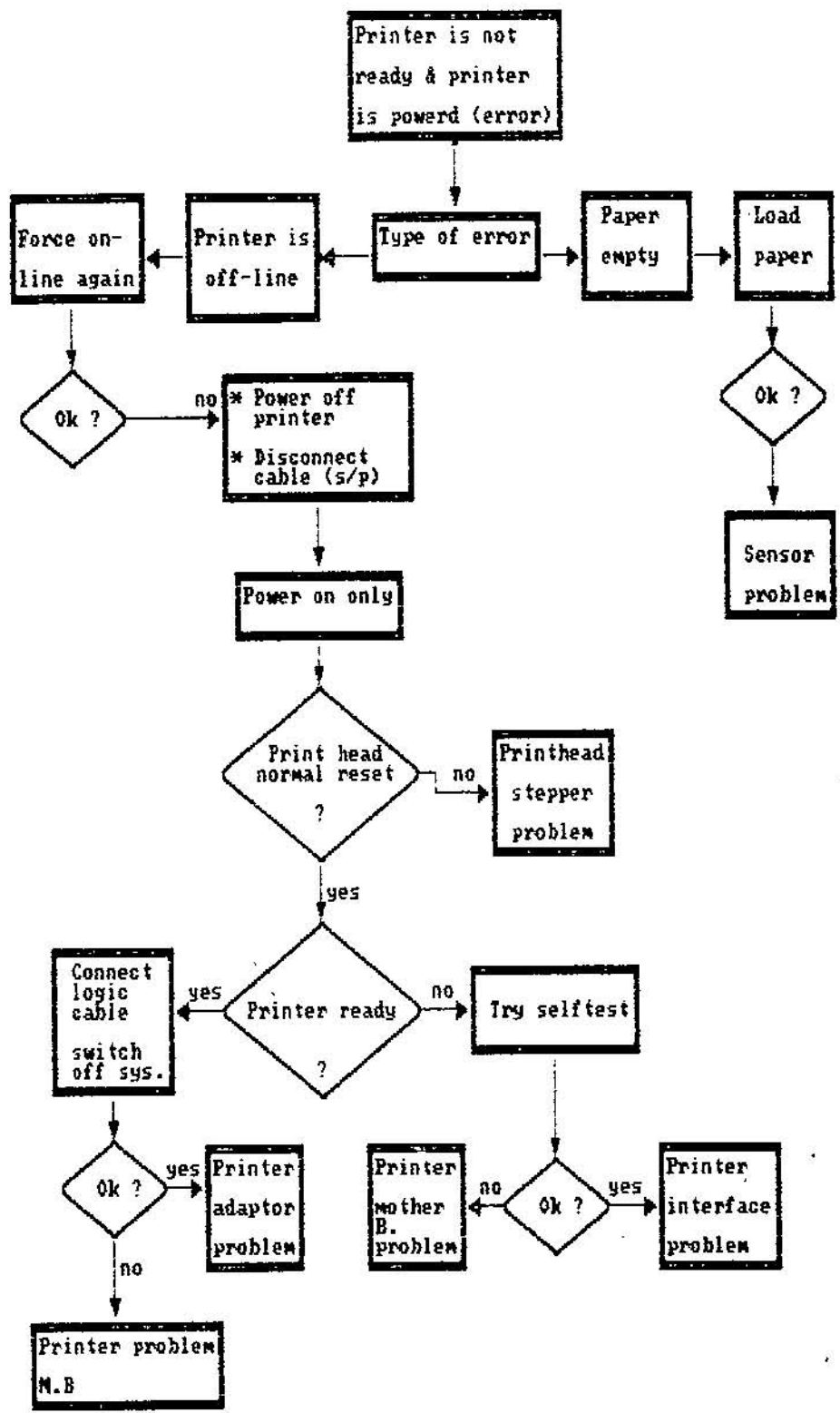


Fig.A.5 Diagnostic flowchart : printer is not ready

**Table B.1 : Symptom = Printout not clear**

- Rule 1: If printout is not clear THEN problem is
  - in ribbon
- Rule 2: If printout is not clear AND ribbon is new THEN problem is in
  - paper thickness lever
- Rule 3: If printout is not clear AND ribbon is OK AND paper thickness lever is OK THEN problem is in
  - print head
- Rule 4: If incomplete characters occur THEN problem is in
  - print head coils, transistors, pins
  - print head cable
- Rule 5: If problem is in print head THEN problem is in
  - print head coils
  - print head transistors (drivers)
  - print head pins

**Table B.2 : Symptom = System Dead**

- Rule 1: If power led is NOT on AND fan is NOT operating THEN problem is
  - in the power source (0.9)
- Rule 2: If fan is operating continuously AND power led is on THEN problem is in
  - system main unit (0.5)
  - one (all) of the peripherals (0.5)
- Rule 3: If all peripherals are disconnected AND power led is on AND fan is operating normally THEN problem is
  - main system unit (0.95)
- Rule 4: If all peripherals are disconnected AND all power source output are disconnected AND fan is NOT operating THEN problem is
  - in power source
- Rule 5: If system is on AND power led is on AND fan is on AND peripherals are connected THEN problem is
  - one (all) of the peripherals (0.5)
  - control cards of peripherals

**Table B.3 : Symptom = No print out**

- Rule 1: If printer could not be ready THEN problem is in
  - cable connection
- Rule 2: If printer is ready AND not print through a specific program THEN problem is in
  - print setup for that program
- Rule 3: If no print through any program AND printer is ready THEN problem is in
  - printer cable connections (0.9)
  - printer adapter
  - printer itself (printer mother board)

- Rule 4: If printer selftest is OK AND configuration is correct AND no print THEN problem is in
- printer cable (0.9)
  - printer adapter (controller)
  - printer interface
- Rule 5: If selftest is NOT OK AND printer is ready THEN problem is in
- printer mother board
- Rule 6: If printer is ready AND selftest is OK AND adapter is OK AND configuration is OK THEN problem is in
- printer interface
- Rule 7: If printer is ready AND adapter is OK AND interface is OK AND selftest is OK AND config. is OK AND head is moving normally THEN problem is in
- head data cable (1.0)

**Table B.4 : Symptom = printer is not ready although powered**

- Rule 1: If printer error is paper empty AND paper is NOT empty THEN problem is in
- printer paper sensor (1.0)
- Rule 2: If printer head doesn't reset normally after power on THEN problem is in
- stopper motor
- Rule 3: If selftest is OK AND printer is NOT ready AND head resets normally THEN problem is in
- printer interface
- Rule 4: If printer is NOT ready AND logical cable is OK AND printer interface is OK THEN problem is in
- printer control card

**Table B.5 : Symptom = monitor malfunctions**

- Rule 1: If monitor is switched on AND screen is not lighting AND power led is NOT lighting THEN problem is in
- monitor power source .
- Rule 2: If monitor is switched on AND screen is NOT lighting AND green LED is ON THEN problem is in
- brightness/contrast control.
- Rule 3: If monitor is switched on AND green LED is lighting AND screen is NOT lighting THEN problem is in
- power source.
  - monitor board.
- Rule 4: If display is not lighting and you hear some hissing sound THEN problem is in
- Horizontal circuit.
- Rule 5: If display is NOT working AND horizontal ckt is OK AND vertical ckt is OK THEN problems in
- monitor mother board.
- Rule 6: If display is lighting AND character size is NOT OK THEN problem is
- horizontal ckt
  - vertical ckt
  - monitor mother board

- Rule 7: If display is lighting AND all colors are NOT THEN  
 problem is in  
 - video board  
 - CRT board
- Rule 8: If screen is half light in a left-right pattern THEN  
 problem is in  
 - horizontal CKT
- Rule 9: If screen is half light in an upper-down pattern THEN  
 problem is in  
 - the vertical CKT
- Rule 10: If characters appear in half patterns AND dip  
 switches are OK THEN problem is in  
 - vertical CKT
- Rule 11: If there is no sound AND screen is NOT OK THEN  
 problem is in  
 - high tension line transformer.
- Rule 12: If characters doesn't appear AND screen is lighting  
 THEN problem is in  
 - data receiver.
- Rule 13: If characters doesn't appear AND screen is lighting  
 AND data receiver is OK THEN  
 - monitor controller  
 - V/H ckts
- Rule 14: If only one central horizontal line appears in the  
 middle of the screen THEN problem is in  
 - vertical ckt
- Rule 15: If only one central vertical line appears in the  
 middle of the screen THEN problem is in  
 - horizontal ckt
- Rule 16: If characters are vibrating THEN problem is in  
 - dip switches  
 - video board
- Rule 17: If the screen is snowy THEN problem is in  
 - power source (not sufficient)

**Table B.6 : Symptom = System does not boot**

- Rule 1: If self test is complete THEN  
 - RAM is OK  
 - ROM is OK  
 - BIOS is OK
- Rule 2: If self test is not complete AND there is no  
 configuration changes THEN problem is in  
 - motherboard (1.0)  
 - most probably in RAM(0.9)
- Rule 3: If system is not booting AND there are error message  
 THEN trigger rules 4,5,6,7,8
- Rule 4: If bad memory indication THEN problem is  
 - in RAM
- Rule 5: If error code is 1701 THEN problem is  
 - in hard disk
- Rule 6: If insert system disk or bad command interpreter THEN  
 problem is in  
 - system diskette

- Rule 7: If run setup program THEN problem is  
 - in the setup of the system
- Rule 8: If setup is certainly correct AND system is still not  
 THEN problem is in  
 - mother board
- Rule 9: If setup program is certainly correct AND system is on  
 after hot booting THEN problem is  
 - in battery
- Rule 10: If error code is 301 THEN problem is  
 - keyboard (0.3)  
 - connection of the keyboard (0.9)  
 - a key hit before selftest is complete (0.8)
- Rule 11: If selftest is complete AND no error messages AND  
 system has a HD AND system boots from floppy disk THEN  
 problem is in  
 - booting area on HD (0.5)
- Rule 12: If selftest is complete AND system is AT AND no error  
 messages AND system doesn't boot from drive A THEN  
 problem is  
 - disk drive controller card (0.9)  
 - HDD + FDD (0.1)
- Rule 13: If selftest is complete AND system is XT AND system  
 is NOT booting from A AND no error messages THEN  
 problem is in  
 - both disk drives
- Rule 14: If system boots from A AND system doesn't boot from C  
 AND system enter the c:\> prompt THEN problem is  
 - in the booting area on the hard disk (1.0)
- Rule 15: If system boots from A AND system doesn't boot from C  
 AND system AND system doesn't enter C THEN problem is  
 - hard disk isn't prepared for DOS partition
- Rule 16: If setup problem always appear although setup  
 procedure is correct THEN problem in mother board
- Rule 17: If system doesn't boot from C AND boots from A AND  
 setup is OK AND mother board is OK THEN problem is  
 - HD tracks (1.0)
- Rule 18: If system loses setup program after powering off THEN  
 problem is  
 - battery (0.8)  
 - mother board
- Rule 19: If system doesn't boot from C AND system boots from A  
 AND transfer system is OK THEN problem is  
 - track 0 on the HD
- Rule 20: If system has two floppy disks AND system NOT boot  
 AND no error messages THEN problem is  
 - floppy disks controller  
 - mother board  
 - diskette drives
- Rule 21: If system doesn't boot AND DOS is reliable AND disk  
 controller is OK THEN problem is  
 - diskette drive (R/W flying heads)
- Rule 22: If DOS is reliable AND FDC is OK AND FDD is OK AND  
 system doesn't boot THEN problem is  
 - mother board

```

**** This program belongs to the deep diagnosis process it
* the problem is in the accessories or in the main modules
** Schedul.prg **

```

```

set talk off
set echo off
set exact on
use decision
scan
delete
pack
endscan
public r1,r2,r3,r4,k
k=0
r1="F"
r2="F"
r3="F"
r4="T"
define window dsn from 5,4 to 10,72 color n/r
use data
old=reccount()
append from component for STATUS=.F.
                                Ins

new=reccount()
k=new-old
skip -k
restore from varlist.mem additive
do while .not. eof()
replace session with num
skip
enddo

use decision
go top
append from component for status=.F.

use decision
go top
scan
if name="mon-pwr-cbl" .or. name="sys-pwr-cbl"
r1="T"
endif
if name="monitor"
r2="T"

endif
if name="prn"
r3="T"
endif
if name="main-sys-unit"
r4="T"
endif
endscan

if r1="T"
activate window dsn
?"      Your system problem is in the computer accesories (like
wait" power leds,fuses...). Press any key to see the decision"

```



```

**** This is the main menu program that displays the main acti
** Main.Prg **
Do while .T.
clear windows
@1,20 say "
@2,20 say " WELCOME TO AL-GHANIM'S EXPERT SYSTEM
DEFINE WINDOW left1 FROM 3,13 To 20,65 COLOR w+/G+
DEFINE WINDOW right1 FROM 18,52 To 22,75 COLOR R+
activate window left1
?"          Here is the Main Menu"
?"=====
?"          Task Type          Task Code "
?"          -----          -"
?"          Edit Knowledge Base          E"
?"
?"          Run a Consultation          R"
?"
?"          Clear working memory          C"
?"
?"          Exit the System          Q"
?"
?"
?"=====
activate window right1
wait"Please enter your TaskCode "to task
do case
case upper(task)="E"
do edit
case upper(task)="R"
do consult
case upper(task)="C"
do clear
Otherwise
return
endcase
enddo

```

```

**** This program adds a new rule ****
** Ruladd.Prg **
clear
clear windows
do While .T.
use rule
kount=reccount()
condno=0
concno=0
set format to rule.fmt
append blank
replace rule_no with kount+1
rulno=rule_no
replace no_of_cond with 0
replace no_of_conc with 0
read
condno=no_of_cond
concno=no_of_conc
use condition
k=1
*do while .T.

kount=reccount()
set format to condition.fmt
append blank
replace rule_no with rulno
replace cond_no with k
replace seriel_no with kount
read
k=k+1
if k=condno+1
exit
endif
enddo

use conclusion
l=1
do while .T.
kount=reccount()
set format to conclusion.fmt
append blank
replace rule_no with rulno
replace conc_no with l

replace seriel_no with kount
read
l=l+1
if l=concno+1
exit
endif
enddo
@l0,1say " "
wait " Do you want to add another rule ? [Y/N] "
```

```
**** This program does the Preliminary diagnosis task ****
** Prelm.Prg **
@1,20 say " Preliminary Diagnosis Session"
define window pr1 from 2,4 to 10,70 color W+/R+
activate window pr1
?" Preliminary diagnosis aims at identifying the main system"
?" module which suffers the malfunction . Generally no exact "
?" specification of the problem diagnosis is given at this stage
?" Now, SWITCH ON your PC system and press any key to"
wait" start the session"
clear windows
do collector
Do interpret
Do scanner
return
```

```

**** This is the main program for editing the knowledge base
** Edit.Prg **
set talk off
set echo off
Do while .T.
clear
clear windows
define window edit from 3,10 to 11,65 color W+/R+
@2,1 say "
@2,24 say " Knowledge Base Edit Menu "
activate window edit
?" Add/Delete/Modify [ Rules ] -----> R"
?" [ Components ] -----> C"
?" [ Symptoms ] -----> S"
?" [ Faults ] -----> F"
?" [ Return ] -----> Q"
wait" Enter your selection "to ch
do case
case upper(ch)="R"
do rule
case upper(ch)="C"

do componen
case upper(ch)="S"
do symp
case upper(ch)="F"
do fault
otherwise
return
Endcase
Enddo

```

```
do collec2m
```

```
endif  
endscan  
endif
```

```
if r3="T"  
scan  
if name="prn"  
do collec2p  
do interp2p  
do scan2p  
do dec2p  
endif  
endscan  
endif
```

```
if r4="T"  
scan  
if name="main-sys-unit"  
do collec2s  
do interp2s  
do scan2s
```

```
do dec2s  
endif  
endscan  
endif  
return
```

```

**** This program collects information for the deep diagnosis session
** Collect1.Prg **
clear windows
@1,1say " "
?" Your system has fault also in one of the main modules"
wait" Press any key to continue .... to part B "
clear
@1,1say " "
?"                               Deep diagnosis session --- B"
define window diagnos from 3,2 to 12,68 color w+/r+
declare array_cond[50]
use condition
go top
k=1
activate window diagnos
scan
if rule_type="monitor"
  array_cond[k]=condition
  i=k
  kk=0
  do while .T.
    if condition=array_cond[i]
      kk=kk+1
      i=i-1
      if i=0
        exit
      endif
    else
      i=i-1
      if i=0
        exit
      endif
    endif
  enddo
  if kk=1
    ?"Adavanced Question #",k
    ?query1
    ?query2
    ?query3
    wait" " to ans
    replace answer with ans
    k=k+1
  endif
endif
endscan
return

```

\*\*\*\* This program displays the accessories problem decision \*\*\*\*

\*\* Decacces.Prg \*\*

```
public kk,k1
clear windows
define window dd from 3,3 to 17,74 color n/g
define window d from 10,2 to 15,72 color n/r
restore from varlist.mem additive
activate window dd
?" Decision Number ",num
?" *****"
?" The following components are likely to be faulty :"
```

---

```
?"Component name", " "Field replacable unit"
?"=====", " "====="
```

```
use decision
go top
scan
?DESCRIB," " ,FRU
endscan
wait
return
```

\*\*\*\* This program performs decision-explanation for the last session

\*\* Explast.Prg \*\*

```
define window amjed from 7,10 to 23,69 color B+/W+
```

```
define window head from 1,3 to 6,77 color g+/w+
```

```
use decision
```

```
go top
```

```
activate window head
```

```
?" Because this Expert System is Rule-Based, the explanation "
```

```
?" given here is not deep . Explanation is based on what the ru
```

```
?" say about a certain problem ."
```

```
?"-----"
```

```
restore from varlist
```

```
activate window amjed
```

```
?"Decision # ",num
```

```
?"====="
```

```
do while .T.
```

```
?describ
```

```
skip
```

```
if eof()
```

```
exit
```

```
endif
```

```
enddo
```

```
go top
```

```
?
```

```
?"Explanation"
```

```
?"====="
```

```
do while .T.
```

```
?becaus1
```

```
?becaus2
```

```
skip
```

```
if eof()
```

```
exit
```

```
endif
```

```

**** This program edits componenet data to the database files ****
** Component.Prg **
Do while .T.
clear
define window com from 4,13 to 10,62 color N+/W+
activate window com
?" Add--A      Delete--D      Modify--M      Return--R"
wait" Your choice is ... "to mv
do case
case upper(mv)="A"
  Do while .T.
  clear windows
  use component
  set format to component.fmt
  append blank
  read
  @18,1 say " "
  wait"          Do you want to add another component record ? [Y/N] "
  If upper(ad)="Y"
    loop
  Endif
  exit
  Enddo
case upper(mv)="D"
  clear windows
  Do while .T.
  clear
  accept"Enter the name of the component that you want to delete "
  use component
  go top
  locate for name=name
  delete
  pack
  wait"This component has been deleted from file, delete another one ?
  ans
  If upper(ans)="Y"
    loop
  endif
  exit
  Enddo
case upper(mv)="M"
  clear windows
  Do while .T.
  clear
  @3,2 say " "
  accept"          Enter the name of the component to be modified "
  use component
  go top
  locate for name=nam
  set format to component.fmt
  read
  @18,1 say " "
  wait"          Do you want to modify another record ? [Y/N]
  If upper(nn)="Y"
    loop
  Endif
  exit
  Enddo
case upper(mv)="R"
  return

```



```

**** This is the main program for editing the knowledge base ****
** Edit.Prg **
set talk off
set echo off
do while .T.
clear
clear windows
define window edit from 3,10 to 11,65 color W+/R+
@2,1 say "
@2,24 say " Knowledge Base Edit Menu "
activate window edit
?" Add/Delete/Modify [ Rules ] -----> R"
?" [ Components ] ----> C"
?" [ Symptoms ] -----> S"
?" [ Faults ] -----> F"
?" [ Return ] -----> Q"
wait" Enter your selection "to ch
do case
  case upper(ch)="R"
    do rule
  case upper(ch)="C"
    do componen
  case upper(ch)="S"
    do symp
  case upper(ch)="F"
    do fault
  otherwise
    return
Endcase
Enddo

```

```

**** This program edits the rule base ***
** Rule.prg **
do while .T.
clear
clear windows
define window rul2 from 1,33 to 13,43 color GR+/W+
define window rul3 from 12,12 to 14,62 color W+/R+
define window rul1 from 1,12 to 3,62 color W+/R+
activate window rul2
?
?"Add A"
?
?"Delete D"
?
?"Modify M"
?
?"Return R"
?
activate window rul1
?" Rule Base Edit Menu"
activate window rul3
wait" Please enter your choice ..... "to chs
do case
  case upper(chs)="A"
    do ruladd
  case upper(chs)="D"
    do ruld1
  case upper(chs)="M"

```

```

**** This program collects information from the user about the
**** of the faulty system ****
** Collector.Prg **

```

```

@1,23 say " Preliminary Diagnosis Session"
define window prl from 2,4 to 10,70 color W+/R+
activate window prl
?" Preliminary diagnosis aims at identifying the main system"
?" module which suffers the malfunction . Generally no exact "
?" specification of the problem diagnosis is given at this stage .
?" Now, SWITCH ON your PC system and press any key to"
wait" start the session"

```

```

public ans1,ans2,ans3,ans4,ans5,ans6,ans7,special
public ans8,ans9,ans10,ans11,ans12,ans13,num

```

```

do while .T.
special=" "

```

```
ans1=" "
```

```
ans2=" "
```

```
ans3=" "
```

```
ans4=" "
```

```
ans5=" "
```

```
ans6=" "
```

```
ans7=" "
```

```
ans8=" "
```

```
ans9=" "
```

```
ans10=" "
```

```
ans11=" "
```

```
ans12=" "
```

```
ans13=" "
```

```
ans14=" "
```

```
ans15=" "
```

```

define window qusn from 11,6 to 20,68 color rg+/W+
Activate window qusn

```

```
k=1
```

```
?"Question #",k
```

```
?"---Do you hear system fan operating : "
```

```
?"A- normally (contineously) ?"
```

```
?"B- operate at start then stops ( unusual operation) ?"
```

```
wait" Chose A or B "to ans1
```

```
k=k+1
```

```
?"Question #",k
```

```
wait"---Is monitor power LED lightening ? [ Y/N ] "to ans2
```

```
If upper(ans2)="Y"
```

```
    k=k+1
```

```
?"Question #",k
```

```
?"---Is monitor operating normally, don't look at the writing "
```

```
wait"now, just watch if it is operating like a flourecent bulb ? "
```

```
If upper(ans3)="Y"
```

```
k=k+1
```

```
?"Question #",k
```

```
wait"---Is there any writing on the screen ? "to ans4
```

```
    If upper(ans4)="N"
```

```
k=k+1
```

```
?"Question #",k
```

```
?"If possible replace the monitor logic cable with another one,"
```

```
wait"do you see any changes on the monitor screen ? "to ans11
```

```

**** This program interprets the data obtained by collec2m ****
** Interp2m.Prg **
use condition
go top
***** Giving the same value for all similar conditions
scan
  nam=condition
  if answer=" "
    exit
  endif
  st=answer
  scan
    if condition=nam
      replace answer with st
    endif
  endscan
endscan
***** Identifying the valid conditions
scan
stat=status
  if answer=stat
    replace valid with .T.
  endif
endscan
***** Finding the rules that are triggered

use rule
do while .not. eof()
x=0
rn=rule_no
nofc=no_of_cond
use condition
c=0
  do while .not. eof()
    if rule_no=rn .and. valid
      c=c+1
    endif
    if eof()
      exit
    else
      skip
    loop
  endif
enddo
close databases
use rule
  locate for rule_no=rn
  if c=nofc
    replace triggered with .T.
    x=x+1
  endif

  if eof()
    exit
  else
    skip
  loop
endif
enddo
close databases
***** Filling the component fields with T/F values *****

```

```
do while .not. eof()
```

```
    rnu=rule_no
    p=no_of_conc
    if triggered
        use conclusion
        go top
        j=0
        do while j<=p
            locate for rule_no=rnu
            naim=conclusion
            y1=why1
            y2=why2
            use component
            locate for name=naim
            replace status with .F.
            replace becaus1 with y1
            replace becaus2 with y2
            j=j+1
            use conclusion
        endif
        if eof()
            exit
        else
            skip
            loop
        endif
    enddo
    use rule
endif
    if eof()
        exit
    else
        skip
        loop
    endif
enddo
return
```

This program scans the component DB. to identify the faulty

```
*** Scan.Prg ***
```

```
use decision  
delete all  
pack
```

```
append from componen for status=.F.
```

```
define window dsn from 10,4 to 15,70 color r+
```

```
use data  
old=reccount()  
append from component for STATUS=.F.  
new=reccount()  
k=new-old  
skip -k
```

```
restore from varlist.mem additive  
do while .not. eof()  
  replace session with num  
  skip  
  if eof()  
    exit  
  endif  
enddo
```

```
activate window dsn  
?" Press any key to see the diagnostic decision "  
wait" that the system has made...."  
return
```

```

**** This program interprets the initial information collected by the
** deep diagnosis programs **
** Interpl.Prg **
set exact on
close databases
use componen
go top
* This part clears the component database and makes them all in True
do while .not. eof()
  if status=.F.
    replace status with .T.
    replace becaus1 with " "
    replace becaus2 with " "
    if eof()
      exit
    else
      skip
      loop
    endif
  endif
  if eof()
    exit
  endif
  skip
  loop
enddo
go top
* rule 1
  If upper(ans8)="N"
  locate for name="main-sys-unit"
  replace status with .F.
  replace becaus1 with "After you change the system power cable, fan
  implying "
  replace becaus2 with "some error in power source or sys. unit"
  Endif
* rule 2
  If upper(ans5)="N"
  locate for name="monitor"
  replace status with .F.
  replace becaus1 with "Since monitor is not showing any response after
  the power cable, "
  replace becaus2 with "this means that monitor unit is not ok or power
  ficient"
  Endif
* rule 3
  If upper(ans8)="Y"
  locate for name="sys-pwr-cbl"
  replace status with .F.
  replace becaus1 with "After changing the power cable, system fan works
  meaning problem is "
  replace becaus2 with "in power cable ( may be end sockets)"
  Endif
* rule 4
  If upper(ans5)="Y"
  locate for name="mon-pwr-cbl"
  replace status with .F.
  replace becaus1 with "After replacing monitor power cable, monitor
  y implying that "
  replace becaus2 with "problem is in monitor pwr cbl"
  Endif
*rule 5

```

- Algorithm** : A procedure that specifies a logical step-wise strategy for solving a specific problem.
- Artificial Intelligence (AI)** : A branch of computer science dedicated to the development of computer "intelligence".
- Backward-Chaining** : The process of starting with a goal state and working backward with an assumption in order to prove the specified goal. Backward-chaining is the reasoning process often used in expert systems.
- Diagnostic System** : An expert system that solves diagnostic problems. Expert systems in this category include automotive diagnostic, general mechanical diagnostic, etc.
- Domain** : The definable boundaries of knowledge relating to a given topic.
- Heuristic** : A technique or structure used in programs to improve the performance of searching for solutions. One common heuristic structure is represented by a tree.
- Heuristic Knowledge** : Knowledge that has a built-in hierarchical or top-down order.
- Inheritance** : The process of deriving values and relationships for objects based on the attributes and values of other objects. Inheritance is often used in frame-based representation systems.
- Knowledge** : A collection of rules, facts, properties, heuristics, and relationships needed to solve a specific problem.
- Knowledge Acquisition** : The science and art of gathering the necessary knowledge to solve a specific problem.
- Knowledge Base** : A collection of all knowledge in the form of rules, facts, relationships, and attributes needed for expert system.
- Knowledge Engineer** : An individual who gathers and represents knowledge for expert system applications. Knowledge engineers work closely with human experts.
- Meta-rule** : A rule that defines other rules.
- Predicate** : A function that can only have a true or false value. Predicates are used to express simple properties about objects.
- Predicate Calculus** : An extension of propositional calculus in which the use of quantified variables are permitted.

**Procedural Knowledge** : Knowledge that can be represented as a procedure or process. Procedural programming languages represent knowledge in the form of algorithms.

**Procedural Language** : A programming language that is based on algorithms. Examples of procedural languages are Pascal, C, BASIC, and FORTRAN.

**Production Rule** : A rule of the IF-THEN format that consists of a premise and conclusion. Production rules are often used in expert systems to represent the knowledge of the expert.

**Proposition** : An expression in which either a true or false statement is made about an object.

**Propositional Calculus** : A formal logic system used to represent the true or false value about objects.

**Prototyping** : The technique of building a preliminary version of a program. Prototyping techniques are often employed by expert system developers.

**Rule** : A clause that expresses relationships between facts.

**Semantic Network** : A general type of knowledge representation in which objects and values are represented as nodes and relations are represented as connections or arcs.

**Symbolic Processing** : The technique of computing with symbols. Symbolic processing is important to artificial intelligence programs because it is believed that people use symbolic processing techniques to solve problems.

**Working Memory** : The memory used in an expert system to store facts and conclusion during the course of a consultation with the system.



## REFERENCES

1. Henry C. Mishkof, ' Understanding Artificial Intelligence ' ,Texas Instruments , (1985) .
2. Jack Krakauer , ' Smart Manufacturing with Artificial Intelligence' , Texas Instruments , (1987) .
3. David king,"Strategies for integrating decision support, data base management, and expert system technology",Expert Systems with Applications, Vol. 1, pp23-38, (1990).
4. Machine Intelligence, Infotech , State of the Art Report , Series 9 , number 3 , (1982) .
5. Keith weiskamp , Terry Hengl , 'Artificial intelligence Programming with Turbo Prolog ', John Wiley & Sons, (1987) .
6. Mahmoud Abu Ali, ' A Systematic Approach to Computer-Aided Maintenance Manegement to IBM PC's and applications, Master Thesis, University of Jordan (1990) .
7. Christopher F. , Chabris , ' Artificial Intelligence and Turbo Pascal ', Oxford Press, (1987) .
8. Engene J. Wittry , ' Managng Information Systems , an Integrated Approach '. Prentice-Hall, (1987) .
9. Cary Kahn and John Mcdermott , 'The Mud Sytem' , IEEE transaction, Expert : intelligent systems and their applications , Vol 1 , No. 1 , Spring (1986) , pp.23-32 .
10. Peter Hart , 'Interveiw ', IEEE transaction , Expert : intelligent systems and their applications , Vol 1 , No. 1 , pp.96-99, spring (1986).
11. M.E. Atwood, R. Brook, E.R. Randlinski,'Causal modele: The next generation of expert systems',Electrical communication,

Vol.60, No.2, pp180-184, (1986).

12. M.E.Atwood, E.R. Randalinski, 'Diagnostic System Architecture', Electrical communication, Vol.60, No.2, pp174-179, (1986) .

13. M.A. Newiead, R. Pettipher, 'Knowledge Acquisition for Expert System', Electrical communication, Vol.60, No.2, pp115-121, (1986).

14. Andew Kusiak, ' Intelligent Manufacturing Systems', Prentice Hall, (1990).

15. Thomas J. Laffey, Walton A. Perkins, and Tin Nguyen, ' Reasoning About Fault Diagnosis with LES', IEEE Expert, Vol.1, No.1, pp13-20, spring (1986).

16. D.Neiman, "Technological Considerations for Industrial Expert Systems', Electical communication, Vol.60, No. 2, pp185-189, (1986).

17. Tean-Louis Lauriere, ' Problem Solving and Artificial Intelligence', Prentice Hall. (1990) .

18. David Bendel Hertz, 'The knowledge Engineering Basis Of Expert Systems', Expert Systems with Applications, Vol.1, pp79-84, (1990).

19. Robert J.Schalkoff, 'Artificial Intelligence, an engineering approach'. McGraw-Hill, (1990).

20. Roy Rada, Paul E. S. Dunne, and Judith Barlow, "EXPERTEXT", Expert Systems with Applications, Vol. 1, pp51-62, (1990).

21. New Patents, Expert Systems with Applications, Vol.1, ppV-VII, (1990).

**399400**

22. Ali R. Uzel, Richard J. Edwards, and Bryan L. Button, "An

expert system for convective heat transfer measurements using transient analysis", Engineering Applications of Artificial Intelligence", Vol.2, No 1, pp40-48, March (1989).

23. Maria Litt, Jack C. H. Chung, David C. Bond and Gary G. Leininger, "A scheduling and planning expert system for multiple furnaces", Engineering Applications of Artificial Intelligence, Vol.1, No 1, pp16-21, March (1988).

24. JAY KYU LEE, SANG BONG OH, and JUNG CHEOL SHIN, 'UNIK-FCST: Knowledge-Assisted Adjustment of Statistical Forecast', Expert Systems with Applications, Vol. 1, pp39-49, 1990.