CIT 474: INTRODUCTION TO EXPERT SYSTEMS

NATIONAL OPEN UNIVERSITY OF NIGERIA

SCHOOL OF SCIENCE AND TECHNOLOGY

COURSE CODE: CIT 474

COURSE TITLE: INTRODUCTION TO EXPERT SYSTEMS

Course Code	**CIT 474**
Course Title	Introduction to Expert Systems
Course Developer/Writer	Olayanju Taiwo Abolaji Computer Department, Federal College of Education (Tech.) Akoka, Lagos
Course Editor	
Programme Leader	
Course Coordinator	

CONTENTS **PAGE**

COURSE GUIDE

Introduction

Introduction to Expert Systems is a First Semester course. It is a two credit degree course available to all four hundred level students offering Information Technology. The course consists of 11 units necessary to acquaint you with the head start on Expert Systems. The understanding from this course will enable you to acquire the skills necessary to develop, operate and maintain software in the areas of expert and intelligent systems. They are no compulsory pre-requisites to it, although it is good to have a basic knowledge of computer software and how it is important in operating a computer system.

What you will learn in this Course

This Course consists of units and a course guide. This course guide tells you briefly what the course is about, what course materials you will be using and how you can work with these materials. In addition, it advocates some general guidelines for the amount of time you are likely to spend on each unit of the course in order to complete it successfully.

It gives you guidance in respect of your Tutor-Marked Assignment which will be made available in the assessment section. There will be regular tutorial classes that are related to the course. It is advisable for you to attend these tutorial sessions. The course will prepare you for challenges you will encounter in the field of Artificial intelligence generally and expert systems in particular.

Course Aim

The aim of the course is simple. The course aims to provide you with an understanding of Expert Systems.

Course Objectives

To achieve the aim set out, the course has a set of objectives which are included at the beginning of each unit. You should read these objectives before you study the unit. You may wish to refer to them during your study to check on your progress. You should always look at the objectives after completion of each unit. By doing so, you would have followed the instruction in the unit. Below are the comprehensive objectives of the course as a whole. By meeting these objectives, you should have achieved the aim of the course as a whole. In addition to the aim above, this course sets to achieve some objectives. Thus, after going through the course, you should be able to understand:

· Basic Concept of Expert Systems
· Components of expert systems, and development of an expert system
· The need for Expert Systems and Applications
· Knowledge Representation in expert systems
· Classes of Expert Systems
· Rule-based expert systems
· Frame-based expert systems
· Fuzzy logic based expert systems
· Neural network based expert systems
· Expert Systems Characteristics and Application
· Natural Language interface for Expert Systems
· Programming Language for Developing Expert System
· Blackboard Expert System – HEARSAY
· Frame Based Expert System-
· Expert System Shells
· Selecting Expert Systems-Based Tool (ESBT) for use in an Organization

· Current Trends in Expert Systems

Working through this Course

To complete this course, you are required to read each study unit, read the textbook and read other materials that may be provided by the National Open University of Nigeria. Each unit contains self-assessment exercises and at certain point in the course, you will be required to complete and submit assignments for assessment purposes. At the end of the course there is a final examination. The course should take you about a total of 17 weeks to complete. Below you will find listed all the components of the course, what you have to do and how you should allocate your time to each unit in order to complete the course on time and successfully. This course entails that you spend a lot of time to read. I would advise that you avail yourself the opportunity of attending the tutorial sessions where you have the opportunity of comparing your knowledge with that of other people.

The Course Materials

The main components of the course are:

· The course Guide
· Study Units
· References/Further Readings
· Assignments
· Presentation Schedule

Study Unit

The study units in this course are as follows:

MODULE 1 Basic Concept of Expert Systems
Unit 1: Introduction to expert systems
Unit 2: Components of expert systems, and development of an expert system
Unit 3: The need for Expert Systems and Applications
Unit 4: Knowledge Representation in expert systems

MODULE 2: Classes of Expert System
Unit 1: A rule-based expert system
Unit 2: Frame-based expert system
Unit 3: Fuzzy and neural network based expert system
Unit 4: Blackboard Expert System – HEARSAY
Unit 5: Expert System Shells

Module 3: Current trends in expert systems.

Each unit consists of one or two week‘s work and includes an introduction, objectives, reading materials, conclusion, and summary, Tutor Marked Assignment (TMAs), references and other resources. The unit directs you to work on exercises related to the required reading. In general, these exercises test you on the materials you have just covered or required you to apply it in some way and thereby assist you to evaluate your progress and to reinforce your comprehension of the material. In addition to TMAs, these exercises will help you in achieving the stated learning objectives of the individual units and of the course as a whole.

Presentation Schedule
Your course materials have important dates for the early and timely completion and submission of your TMAs and attending tutorials. You should remember that you are required to submit all your assignments by the stipulated time and date. You should guard against falling behind in your work.

Assessment
There are three aspects to the assessment of the course. First is made up of self-assessment exercises, second consists of the Tutor-Marked Assignments and third is the written examination/end of course examination. You are advised to do the exercises. In tackling the assignments, you are expected to apply information, knowledge and techniques you gathered during the course. The assignments must be submitted to your facilitator for formal assessments in accordance with the deadlines stated in the presentation schedule and the assignment file. The work you submit to your tutor for assessment will count for 30% of your total course work. At the end of the course you will need to sit for a final or end of course examination of about three hour duration. This examination will count for 70% of your total course mark.

Tutor-Marked Assignment
The TMA is a continuous assessment component of your course. It accounts for 30 % of the total score. You will be given four (4) TMAs to answer. Three of these must be answered before you are allowed to sit for the end of course examination. The TMAs would be given to you by your facilitator and returned after you have done the assignment. Assignment questions for the units in this course are contained in the assignment file. You will be able to complete your assignment from the information and the material contained in your reading, references and the study units. However, it is desirable in all degree level of education to demonstrate that you have read and researched more into your references, which will give you a wider view point and may provide you with a deeper understanding of the subject. Make sure that each assignment reaches your facilitator on or before the deadline given in the presentation schedule and assignment file. If for any reason you cannot complete your work on time, contact your facilitator before the assignment is due to discuss the possibility of an extension. Extension will not be granted after the due date unless there are exceptional circumstances.

Final Examination and Grading
The end of your examination for Introduction to Expert Systems will be for about 3 hours and it has a value of 70% of the total course work. The examination will consist of questions, which will reflect the type of self-testing, practice exercise and tutor marked assignment problems you are previously encountered. All areas of the course will be assessed. You are to use the time between finishing the last unit and sitting for the examination to revise the whole course. You might find it useful to review your self-test, TMAs and comments on them before the examination. The end of course examination covers information from all parts of the course.

Course Marking Scheme

Assignment	**Marks**
Assignment 1-4	Four assignments, best three marks of the four count at 10% each- 30% of course marks
End of course examination	70% of overall course marks
Total	100% of course materials

Facilitator/Tutor and Tutorials

There are 16 hours of tutorials provided in support of the course. You will be notified of the dates, times and location of these tutorials as well as the name and phone number of your facilitator, as soon as you as you are allocated a tutorial group. Your facilitator will mark and comment on your assignments, keep a close watch on your progress and any difficulties you might face and provide assistance to you during the course. You are expected to mail you Tutor Marked Assignment to your facilitator before the schedule date. (at least two working days are required). They will be marked by your tutor and returned to you as soon as possible. Do not delay to contact your facilitator by telephone or e-mail if you need assistance The following might be the circumstances in which you would find assistance necessary, you would have to contact your facilitator if :

· Understand any part of the study or assigned reading
· You have difficulty with the self- tests
· You have a question or problem with an assignment or with the grading of an assignment You should endeavor to attend the tutorials. This is the only chance to have face to face contact with your course facilitator and to ask question which are answered instantly. You can raise any problem encountered in the course of your study. To gain much benefit from the course tutorials, prepare a question list before attending them. You will learn a lot from participating actively in the discussions.

Summary

Introduction to Expert Systems is a course that intends to provide concept of the discipline and is concerned with finding solutions through the software system on the basis of expert knowledge or provides an evaluation of known problems. Upon the completion of the course, you will be equipped with the knowledge of expert system and the application of its reasoning capabilities to reach a conclusion. You will be exposed to various application areas of expert systems. Furthermore, you will be able to answer the following types of questions:

· What is Expert System?
· What are roles of individuals who interact with expert system?
· What are various application areas of expert systems?

Of course a lot more question you will be able to answer.

I wish you success in the course and I hope you will find it both interesting and useful.

MODULE 1: Basic Concept of

Expert Systems

Unit 1: Introduction to Expert

Systems

1.0 Objectives

By the end of this unit, you should be able to:

- Understand the historical background of expert system
- Define an expert system
- Identify the human factors who develop and interact with expert system
- List the advantages and disadvantages of expert system
- State the features of expert system.

1.1 Background and History

Expert systems (ES) are systems that emanate from the new area of computing known as Artificial Intelligence (AI). AI is the branch of Computer Science concerned with developing computers that behaves like humans. Precisely, Expert systems occupy a central place in the cognitive science aspect of artificial intelligence as shown in figure 1 below.

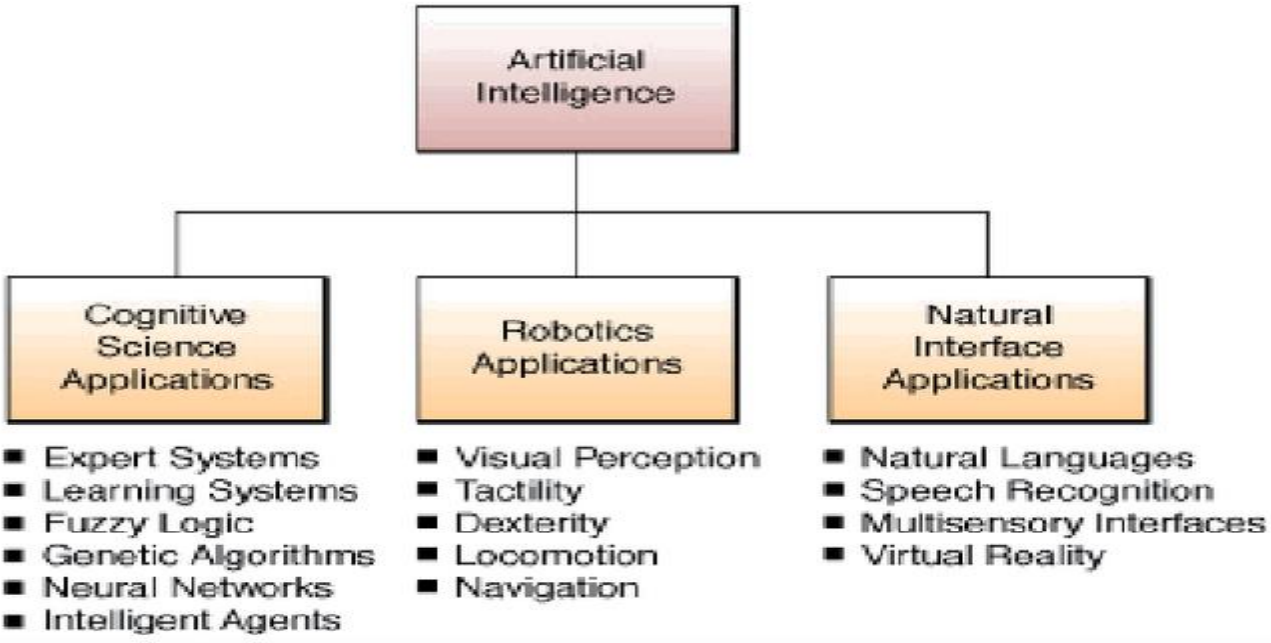

Figure 1: Showing the place of expert system in the broad AI domain

In the mid-1960s, Edward A. Feigenbaum was one of the people in artificial intelligence research who decided, that it was expedient to know how much a computer program can know and that the best way to find out would be to try to construct an artificial expert. In the course of looking for an appropriate field of expertise, Feigenbaum collaborated with Joshua Lederberg, the Nobel laureate biochemist, who then suggested that organic chemists sorely needed assistance in determining the molecular structure of chemical compounds.

In 1965, Lederberg and Feigenbaum together with Bruce Buchanan, according to the requirements of the National Aeronautics and Space Administration, at the Stanford University started work on Dendral, the first expert system, at Stanford University. This project started because the conventional computer-based systems had failed to provide organic chemists with a tool for forecasting molecular structure. Human chemists know that the possible structure of any chemical compound depends on a number of basic rules about how different atoms can bond to one another. They also know a lot of facts about different atoms in known compounds. When they make or discover a previously unknown compound, they can gather evidence about the compound by analyzing the substance with a mass spectroscope-which provides a lot of data, but no clues to what it all means.

The DENDRAL system can automatically generate molecular structures that can interpret spectral data.

The program is a first successful program that uses the knowledge of the problem itself rather than the

complex search technology. The DENDRAL guides AI and expert system researchers to realize that the

intelligent behavior relies not only on interference methods but also on the knowledge used in their

interference. Researchers begin to build the program that uses the rules of the code to represent the

knowledge to solve the input problem.

"Knowledge engineering" is the art, craft and science of observing human experts, building models of their expertise and refining the model until the human experts agree that it works. One of the first spinoffs from Dendral was Meta-Dendral, an expert system for those people whose expertise lies in building expert systems. By separating the inference engine from the body of factual knowledge, Buchanan was able to produce a tool for expert-systems builders.

Since then, MIT has also developed the MACSYMA system. The MACYSMA was a mathematician's assistant, which uses heuristics to transform algebraic expressions. After continuous expansion, it can solve more than 600 kinds of mathematical problems, including calculus, matrix operation, solving equation, etc. The successful development of these systems makes the expert system widely concerned by academia and

Engineering. Many researchers in the process of developing expert systems have realized that knowledge representation, knowledge utilization and knowledge acquisition are three basic issues of artificial intelligence systems.

In the mid-1970s MYCIN was developed by Edward H. Shortliffe, a physician and computer scientist at Stanford Medical School. The problems associated with diagnosing a certain class of brain infections was an appropriate area for expert system research and an area of particularly pressing human need because the first twenty-four to forty-eight hours are critical if the treatment of these illnesses is to succeed. With all its promise, and all its frightening ethical implications, medicine appears to be one of the most active areas of application for commercial knowledge engineering.

MYCIN's inference engine, known as E-MYCIN, was used by researchers at Stanford and Pacific Medical Center to produce Puff, an expert system that assists in diagnosing certain lung disorders. An even newer system, Caduceus, now has a knowledge base-larger than any doctor's- of raw data comprising about 80 percent of the world's medical literature.

Prospector, developed by SRI International, looks at geological data instead of molecules or symptoms. Recently this program accurately predicted the location of a molybdenum deposit that may be worth tens of millions of dollars.

In the late 1980s, the Framework-Based Expert System began to enter people's vision. Because of its higher ability to represent descriptive and behavioral object information, a Framework-Based Expert System could handle more complex problems than the Rule-Based Expert System. At the same time, the study of the expert system encountered difficulties, exposing the defects of artificial intelligence systems, such as narrow application areas, knowledge acquisition difficulties, reasoning mechanism and so on. Researchers need to get rid of the dilemma by exploring the basic point of view and the use of new techniques and theories.

1943	Post production rules: McCulloch and Pitts Neuron model
1954	Markov algorithm for controlling rule execution
1956	Dartmouth conference
1957	Perceptron by Rosenblatt, GPS by Newell, Shaw and Simon
1958	LISP by McCarthy
1962	Rosenblatt‘s principle of neurodynamics
1965	Resolution for automated theorem proving by Robinson Fuzzy logic by Zadeh
	DENDRAL (1st ES) designed by Feigenbaum and Buchanan
1968	Semantic nets and associative memory by Quillian

1969	MACSYMA: expert system in mathematics by Martin and Moses
1970	PROLOG by Colmerauer and Roussel
1971	HEARSAY I (speech recognition)
1973	MYCIN (medicine) by Shortliffe and al., followed by GUIDON (tutoring) by Clancey TEIRESIAS (explanation) by Davis EMYCIN (shell) by Van Melle, Shortliffe and Buchanan HEARSAY II (blackboard: multiple expert cooperation)
1975	Frames by Minsky
1976	AM (artificial Math.):creative discovery of math concepts by Lenat Dempster-Shafer theory of evidence for reasoning under uncertainty PROSPECTOR (mineral exploration) by Duda and Hart
1977	OPS expert system Shell by Forgy, used in XCON/R1
1978	XCON/R1 (DEC compzter config.) by McDermott Meta-Dendral (meta-rules and -induction) by Buchanan History (cont.)
1979	Rete algorithm for efficient pattern matching AI becomes commercial Inference Corp. Formed (releases ART expert system in 85)
1980	Symbolics (-> LISP machines)
1982	SMP math expert system Hopfield neural net Fifth generation project in Japan
1983	KEE expert system tool by Intellicorp
1985	CLIPS expert system tool by NASA

Expert system is referred to as a system systems that that the ability to emulates the cognitive skills of human experts to guide users thorough complex decision-making processes.

In the 1970s, the advances in computing have reconsidered that to make the computer to solve an intellectual problem one had to know the solution. In other words, one has to have the know-how in some specific domain as expert in that area would solve the problem. The functionalities of these types of systems are based on knowledge of its task and logical rules or procedures obtained from the experience of a specialist in the area.

This system uses a knowledge base, which is carefully formulated on the basis of expert judgment, intuition, and experience.

About two dozen corporations are currently selling expert systems and services.

Teknowledge, founded by Feigenbaum and associates in 1981, was the first. IntelliGenetics is perhaps the most exotic, specializing in expert systems for the genetic engineering industry. Start-ups in this field tend toward science-fiction names-Machine Intelligence Corporation, Computer Thought Corp., Symbolics, etc. Other companies already established in non-AI areas have entered the field-among them, Xerox, DEC, IBM, Texas Instruments and Schlumberger.

Expert systems are now in commercial and research use in a number of fields:

- KAS (Knowledge Acquisition System) and Teiresias help knowledge engineers build expert systems.
- ONCOCIN assists physicians in managing complex drug regimens for treating cancer patients.
- Molgen helps molecular biologists in planning DNA experiments.
- Guidon is an education expert that teaches students by correcting answers to technical questions.
- Genesis assists scientists in planning cloning experiments.
- TATR is used by the Air Force in planning attacks on enemy air bases.

1.2 What are Expert Systems?

Various definitions of expert systems have been offered by several authors:

- An expert system belongs to a field of artificial intelligence, and it is a computer program that simulates the judgment and behavior of an individual that has expert knowledge and experience in a particular field. It is a knowledge-based computer program that exhibits a degree of expertise in a particular domain thereby solving problem or making decisions that is comparable to that of a human expert.
- It could also be referred to as an AI programs that achieve expert-level competence in solving problems in task areas by bringing to bear a body of knowledge about specific tasks. It could be referred to as *knowledge-based* or *expert systems*
- A type of application program that makes decisions or solves problems in a particular field by using knowledge and analytical rules defined by experts in the field.
- Expert systems represent a branch of artificial intelligence, aiming to take the experience of human specialists and to transfer to a computer system. Specialty knowledge is stored in the computer, which by an execution system (inference engine) is reasoning and
  it derives specific conclusions for the problem.
- A computer program that uses knowledge and reasoning techniques to solve problems that normally require the abilities of human experts. Software that applies human-like reasoning involving rules and heuristics to solve a problem.
- An Expert System (EPS) is a software system, which finds solutions on

the basis of expert knowledge or provides an evaluation of known problems. Examples are systems for the support of medical diagnosis or for the analysis of scientific data.

- Is a particular development of Artificial intelligence that helps to solve problems or make decisions through the use of a store of relevant Information (known as the Knowledge base, and derived from one or more human experts), and a set of reasoning techniques.
- Artificial intelligence based system that converts the knowledge of an expert in a specific subject into a software code. This code can be merged with other such codes (based on the knowledge of other experts) and used for answering questions (queries) submitted through a computer.
- Despite these definitions of Expert systems by different authors, it is important to summarize that an expert system is a knowledge-based computer program that exhibits, within a specific domain, a degree of expertise in problem solving that is comparable to that of a human expert. This problem method uses a knowledge base, which is carefully formulated on the basis of expert judgment, intuition, and experience. Thus an expert system embodies the cognition and ability of an expert in a certain realm, thus emulating the decision-making ability of a human.

The term expert system is reserved for programs whose knowledge base contains the knowledge used by human experts, in contrast to knowledge gathered from textbooks or non-experts. The two terms, expert systems (ES) and knowledge-based systems (KBS), are sometimes used synonymously. Taken together, they represent the most widespread type of AI application. The area of human intellectual endeavor to be captured in an expert system is called the *task domain*. *Task* refers to some goal-oriented, problem-solving activity. *Domain* refers to the area within which the task is being performed. Typical tasks are diagnosis, planning, scheduling, configuration and design. An example of a task domain is aircraft crew scheduling.

In most cases, the purpose of the expert systems is to help and support user's reasoning and not to replace human expert judgment. In fact, expert systems offer to the inexperienced user a solution when human experts are not available.

Expert system is a subset of intelligent systems design under AI, however, expert and intelligence are not the same thing. Let us try to differentiate between intelligence and expertise with a view to uncovering the focal area of expert system in the broad discussions of intelligent systems

1.3 Expert systems and conventional programs

Expert systems are different from traditional application programs in that their capability to deal with challenging real world problems through the application process that reflect human judgment and intuition.

Expert systems should not be confused with cognitive modeling programs, which attempt to simulate human mental architecture in detail. Expert systems are practical programs that use heuristic strategies developed to solve specific classes of problems.

Expert Systems Applications	**Conventional Systems Applications**
Knowledge is fragmented and implicit, is difficult to communicate except in small "chunks", and is often distributed amongst individuals who may disagree.	Knowledge is complete and explicit, and is easily communicated with formulas and algorithms.
Rules are complex, conditional and often defined as imprecise "rules of thumb".	Rules are simple with few conditions.
The finished system captures, distributes and leverages expertise	The finished product automates manual procedures
Problem-solving demands dynamic, context-driven application of facts, relationship and rules	Problem-solving requires predictable and repetetive sequences of actions.
System performance is measured in degrees of accuracy and completeness where explanations may be required to establish correctness.	Simple criteria are used to determine accuracy and completeness.
Inferencing	Program flow
Knowledge based	Database
Object class	Relational class

1.4 The Role of Heuristic Knowledge in expert systems

Much of the knowledge of domain experts in solving practical problems consists of heuristics acquired through learning and experience. A heuristic is a rule of thumb, fact, or even a procedure that can be used to solve some problem, but it is not guaranteed to do so. It may fail. Heuristics can be conveniently regarded as simplifications of comprehensive formal descriptions of a real-world system. For example, it is conceivable that all aspects of the operation of a machine could be completely described in a complex physical or mathematical model, including circumstances under which machines malfunction. In principle, this model could be used to analyze machine problems and (algorithmically) determine malfunctions with virtual certainty. In practice, complete models are often difficult to develop due to lack of necessary information about the problem and its inherent complexity. Therefore, for many problems, domain experts find it practical and necessary to substitute heuristic knowledge for a complex model. Expert system systems benefit from heuristic principles.

1.5 Elements of an Expert System

Expert systems store expert knowledge and apply it "on demand" to solve problems. Most often the user of an expert system is a person. The user may also be another software system or even a mechanical device. A human user, known as an end user usually provides information to the expert system via a computer terminal. The expert system uses inference procedures to apply its

stored knowledge to the facts describing a problem. The systematic application of inference leads to solutions that are then displayed at the terminal.
The operation of an expert system can be viewed in terms of the interaction of distinct components. The knowledge base stores knowledge about how to solve problems. Inference procedures are executed by a software module called the inference engine. If the user of the expert system is a person, communications with the end-user are handled via an end user interface.

Each of the major parts of the expert system architecture can be further explain as below:

1.5.1 The Knowledge Base

Knowledge is stored in the knowledge base using symbols and data structures to stand for important concepts. The symbols and data structures are said to represent knowledge. Knowledge representation can take many forms. The most common form is the production rule. Production rules are a particularly convenient way of expressing heuristic knowledge. The knowledge base refers to the actual store of knowledge for a particular expert system.

A knowledge representation system may be simple, consisting only of data structures for representing rules. Or knowledge representation may incorporate other more complex structures. Knowledge represented in data structures, such as rules, is said to be stated declaratively.

Declarative knowledge is knowledge that is stated explicitly and is intended to be accessible to persons who may need to see it, such as domain experts. The ability to make its declarative knowledge accessible and understandable is one of the most important services provided by a knowledge representation system.

1.5.2 The Inference Engine

The inference engine is a software module that executes procedures for applying knowledge to produce new information about a problem. In production rule systems, an inference engine compares rules against known facts in the context file to determine if new facts can be inferred. The conditions in the premise, or IF part, of the production rules are compared against known facts. If these conditions are satisfied, the facts in the conclusion, or
THEN part, can be inferred. The newly concluded facts are then added to the context file of the expert system.

1.5.3 Expert System Interfaces

Expert systems communicate with human users as well as other software and hardware systems. Expert systems communicate with human users via an end user interface. The purpose of the end user interface is to obtain information about the problem from the end user and to display solutions. To obtain information, the interface may display questions at a terminal and prompt the end user for answers. Solutions may consist of text statements. More elaborate end user interfaces may use graphics and hypertext.
A useful function of an expert system is the ability to explain its actions. While using an expert system, the end user may wish to know why questions are being asked or why certain facts were concluded. When the solution is displayed to the end user, the user may request an explanation of

how the solution was reached. The end user interface contains procedures that generate explanations that can be shown to the end user.
In many practical applications, an expert system must interface, and exchange data, with other software and hardware systems. The number of expert systems that have non-human users, such as other software systems or electronic process control devices, is increasing. Therefore, it is important to note that the expert system interface with both end-user and other non-human components.

1.6 The Human Elements in Expert Systems development

Building an expert system is known as ***knowledge engineering*** and its practitioners are called *knowledge engineers*. The knowledge engineer must make sure that the computer has all the knowledge needed to solve a problem.

Moreover, it is important to identify different kinds of people who are needed to develop and use an expert system and what skills are needed. In general, there are five members of the expert system development team: the domain expert, the knowledge engineer, the programmer, the project manager and the end-user. The success of the expert system entirely depends on how well the members work together.

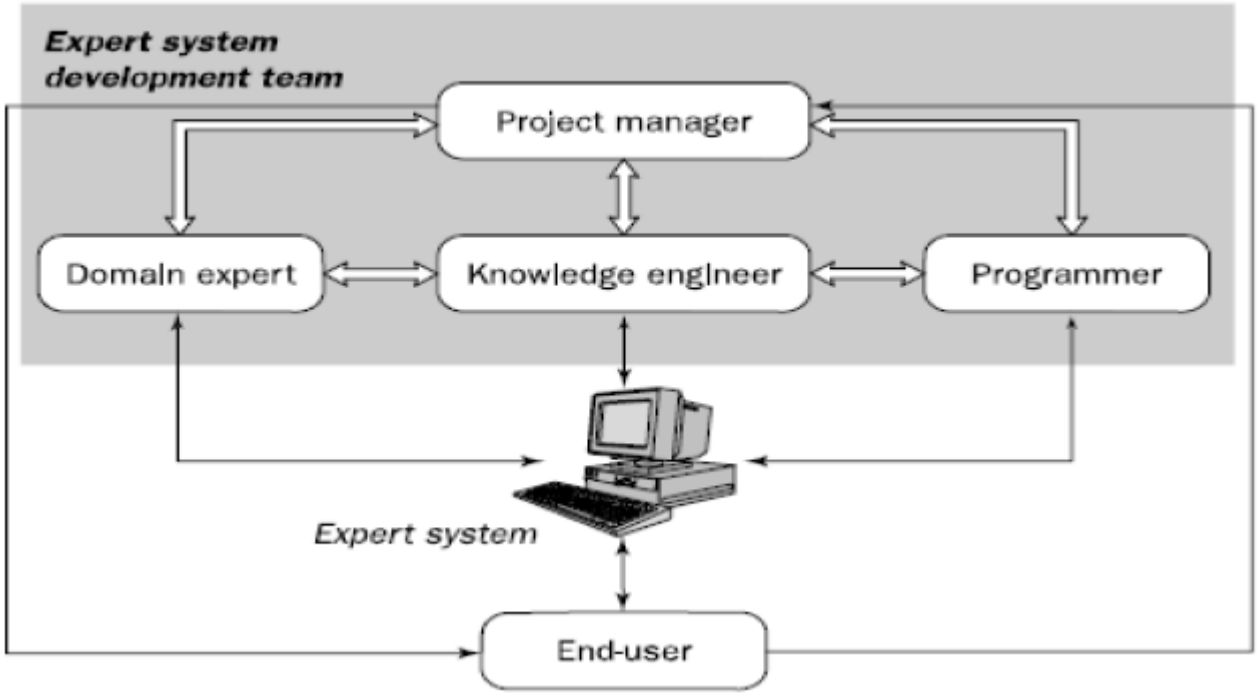

Figure 2: Showing the human factors in the expert system development teams

- **Domain expert:** Anyone can be considered a domain expert if he or she has deep knowledge (of both facts and rules) and strong practical experience in a particular domain. In general, an expert is a skillful person who can do things other people cannot. Consider several

Human Expert examples: A doctor, Chess grands–master, Financial wizard, A chef, an Engineer, an architect, a pharmacist e.t.c

- **Knowledge engineers**: The individual who encodes the expert' knowledge in a declarative form that can be used by the expert system. They are also people who are capable of designing, building and testing an expert system. This person is responsible for selecting an appropriate task for the expert system. He or she interviews the domain expert to find out how a particular problem is solved. Through interaction with the expert, the knowledge engineer establishes what reasoning methods the expert uses to handle facts and rules and decides how to represent them in the expert system. The knowledge engineer then chooses some development software or an expert system shell, or looks at programming languages for encoding the knowledge (and sometimes encodes it himself). And finally, the knowledge engineer is responsible for testing, revising and integrating the expert system into the workplace. Thus, the knowledge engineer is committed to the project from the initial design stage to the final delivery of the system, and even after the project is completed, he or she may also be involved in maintaining the system.

  − Usually the System Builder

- **The programmer**: is the person responsible for the actual programming, describing the domain knowledge in terms that a computer can understand. The programmer needs to have skills in symbolic programming in such AI languages as LISP, and Prolog and also some experience in the application of different types of expert system shells. In addition, the programmer should know conventional programming languages like Python, R, Java, Julia, C, Pascal, FORTRAN e.t.c. If an expert system shell is used, the knowledge engineer can easily encode the knowledge into the expert system and thus eliminate the need for the programmer. However, if a shell cannot be used, a programmer must develop the knowledge and data representation structures (knowledge base and database), control structure (inference engine) and dialogue structure (user interface). The programmer may also be involved in testing the expert system.

- **The project manager:** is the leader of the expert system development team, responsible for keeping the project on track. He or she makes sure that all deliverables and milestones are met, interacts with the expert, knowledge engineer, programmer and end-user.

- **User:** often called just the end-user, is an individual who will be consulting with the system to get advice which would have been provided by the expert. He or she is a person who uses the expert system when it is developed. The user might be an analytical chemist determining the molecular structure of soil from Mars, a junior doctor diagnosing an infectious blood disease, an exploration geologist trying to discover a new mineral deposit, or a power system operator needing advice in an emergency. Each of these users of expert systems has different needs, which the system must meet: the system's final acceptance will depend on the user's satisfaction. The user must not only be confident in the expert system performance but also feel comfortable using it. Therefore, the design of the user interface of the expert system is also vital for the project's success; the end-user's contribution here can be crucial.

− Possible Classes of Users

-A non-expert client seeking direct advice - the ES acts as a Consultant or Advisor

-A student who wants to learn

-An ES builder improving or increasing the knowledge base – a Partner

-An expert - a Colleague or Assistant

It is important to note that the development of an expert system can be started when all five players have joined the team. However, many expert systems are now developed on personal computers using expert system shells. This can eliminate the need for the programmer and also might reduce the role of the knowledge engineer. For small expert systems, the project manager, knowledge engineer, programmer and even the expert could be the same person. But all team players are required when large expert systems are developed.

1.7 Features of Expert Systems

Expert system operates as an interactive system that responds to questions, asks for clarifications, makes recommendations and generally aids the decision-making process.
Expert systems have many features:

- Operates as an interactive system This means an expert system :
  - Responds to questions
  - Asks for clarifications
  - Makes recommendations
  - Aids the decision-making process.

- Tools have ability to sift (filter) knowledge
  - Storage and retrieval of knowledge
  - Mechanisms to expand and update knowledge base on a continuing basis.

- Make logical inferences based on knowledge stored
  - Simple reasoning mechanisms is used
  - Knowledge base must have means of exploiting the knowledge stored, else it is useless; e.g., learning all the words in a language, without knowing how to combine those words to form a meaningful sentence.

- Ability to Explain Reasoning
  - Remembers logical chain of reasoning; therefore user may ask
    1. for explanation of a recommendation
    2. factors considered in recommendation
  - Enhances user confidence in recommendation and acceptance of expert system

- Domain-Specific

- A particular system caters a narrow area of specialization; e.g., a medical expert system
- Quality of advice offered by an expert system is dependent on the amount of knowle

■ Capability to assign Confidence Values
- Can deliver quantitative information
- Can interpret qualitatively derived values
- Can address imprecise and incomplete data through assignment of confidence values.

■ Cost-Effective alternative to Human Expert
- Expert systems have become increasingly popular because of their specialization, albeit in a narrow field.
- Encoding and storing the domain-specific knowledge is economic process due to small size.
- Specialists in many areas are rare and the cost of consulting them is high; an expert system of those areas can be useful and cost-effective alternative in the long run.

■ Goal Driven Reasoning or Backward Chaining
An inference technique which uses IF-THEN rules to repetitively break a goal into smaller sub-goals which are easier to prove;

■ Coping with Uncertainty
The ability of the system to reason with rules and data which decision to make.
Often the Knowledge is imperfect which causes uncertainty.
To work in the real world, Expert systems must be able to deal with uncertainty.
- One simple way is to associate a numeric value with each piece of information in the system.
- the numeric value represents the certainty with which the information is known.

There are different ways in which these numbers can be defined, and how they are combined during the inference process.

■ Data Driven Reasoning or Forward Chaining
An inference technique which uses IF-THEN rules to deduce a problem solution from initial data;

■ Data Representation
Data representation is the way in which the problem specific data in the system is stored and accessed. Expert system is built around a knowledge base module.

- Knowledge acquisition is transferring knowledge from human expert to computer.
- Knowledge representation is faithful representation of what the expert knows.

No single knowledge representation system is optimal for all applications.

The success of expert system depends on choosing knowledge encoding scheme best for the kind of knowledge the system is based on.

The IF-THEN rules, Semantic networks, and Frames are the most commonly used representation schemes.

■ User Interface

The user interface is the portion of the system which creates an intermediary between the system and the user for interaction. The acceptability of an expert system depends largely on the quality of the user interface.

- Scrolling dialog interface: It is easiest to implement and communicate with the user.
- Pop-up menus, windows, mice are more advanced interfaces and powerful tools for communicating with the user; they provide graphic support.

■ Explanations

The ability of the system to explain the reasoning process that it used to reach a recommendation. An important feature of expert systems is their ability to explain themselves. Given that the system knows which rules were used during the inference process, the system can provide those rules to the user as means for explaining the results.

By looking at explanations, the knowledge engineer can see how the system is behaving, and how the rules and data are interacting. This is very valuable diagnostic tool during development.

1.8 Benefits/advantages of Expert Systems

- Permanence: Expert systems do not forget.
- Reproducibility: Copies of an expert system can be made.
- Power: For applications where there is a maze of rules exhibited, it can be unraveled by the expert system.
- Efficiency: Expert systems can increase throughput and reduce personnel costs.
  -Expert systems are inexpensive to operate.
  -Development costs can be amortized over many years.
  - Expert systems can eliminate routine costs and reduce major maintenance costs.
- Consistency: With expert systems, similar events are handled the same way. Expert systems will make comparable recommendations for 'like' situations and are not affected by recent or primary effects.
- Documentation: Expert systems provide permanent documentation of the decision process.

- Completeness: An expert system can review all the transactions or possibilities.
- Timeliness: Fraud and/or errors can be prevented. Information is available sooner for decision making and action. The expert system works 24 hours a day, all year long.
- Scope: The expert system can encompass the cumulative expertise of many human experts.
- Business success: Owners reduce the inherent risks of conducting their business due to:
    - Consistency of decision making.
    - Documentation (ISO requirements)
    - Acquired expertise
- Positive impacts:
    - Productivity gains and cost savings.
    - Critical new tool for managers and a proactive answer to expertise attrition.
    - Decisions and solutions are more consistent and less subject to biases or sensitivity to the environment
    - Employment: shift to-wards more satisfying work.

1.9 Disadvantages of Expert Systems

- Common sense: In addition to a great deal of technical knowledge, human experts have commonsense. It is not yet known how to give expert systems common sense.
-Creativity: Human experts can respond creatively to unusual situations, expert systems cannot.
- Learning: Human experts automatically adapt to changing environments; expert systems must explicitly update. Case-based reasoning and neural networks are methods that can incorporate learning.
-Sensory experience: Human experts have available to them a wide range of sensory experience; expert systems are currently dependent on symbolic input.
-Degradation: Expert systems are not good at recognizing when no answer exists or when the problem is outside their area of expertise.

Activity A: What is an Expert System?

Conclusion

This Unit has introduced you to the historical background of expert systems, concept of expert system, differences between expert system and conventional programs, the elements of expert systems, its features and components as well as advantages and disadvantages of expert systems. You should have been able to also identify the various individuals who interact with expert system.

Summary

In this Unit we have learnt that:

- **An expert system** is as a knowledge-based computer program that exhibits, within a specific domain, a degree of expertise in problem solving that is comparable to

that of a human expert. This problem method uses a knowledge base, which is carefully formulated on the basis of expert judgment, intuition, and experience.

- **Domain expert:** is an individuals who currently is expert in solving the problems; the system is intended to solve
- **Knowledge engineer**: is individual who encodes the expert's knowledge in a declarative form that can be used by the expert system.

- **User:** is the individual who will be consulting with the system to get advice which would have been provided by the expert.
- **System engineer** – is the individual who builds the user interface, designs the declarative format of the knowledge base, and implements the inference engine

Tutor Marked Assignment

- Highlight the historical development of expert system
- Define an expert system
- What is the relationship between ES and AI?
- List and explain the elements of expert systems
- Briefly state the role of individuals who interact with expert system
- List the advantages and disadvantages of expert system
- State the features of expert system.

Further Readings and Other Resources

- Giarratano, J. (2004). Expert Systems: Principles and Programming (4th Edition), Thomson Course Technology: Gary Riley
- CLIPS User's Guide
- Jackson, P. (1999). Introduction *to Expert Systems (*3rd edition), Addison Wesley Longman, Harlow, England
- David S. Prerau, Developing *and Managing Expert Systems*, John Durkin, Expert Systems: Catalog of Applications,

Unit 2: components of expert systems, and development of an expert system

1.0 Introduction

This unit highlights the components of expert systems, expert system in operation, as well as various steps in developing expert system.

2.0 Objectives

By the end of this unit, you will be able to:

- Discus the components of expert system
- Understand expert system in operation
- Explain the various steps of developing an expert system

3.0 Components of an Expert System:

Expert systems are usually built for specific application areas called Domain. The components of Expert System are as follows:

- User Interface.
- Inference Engine.
- Knowledge base.
- Working memory.
- Explanation facility

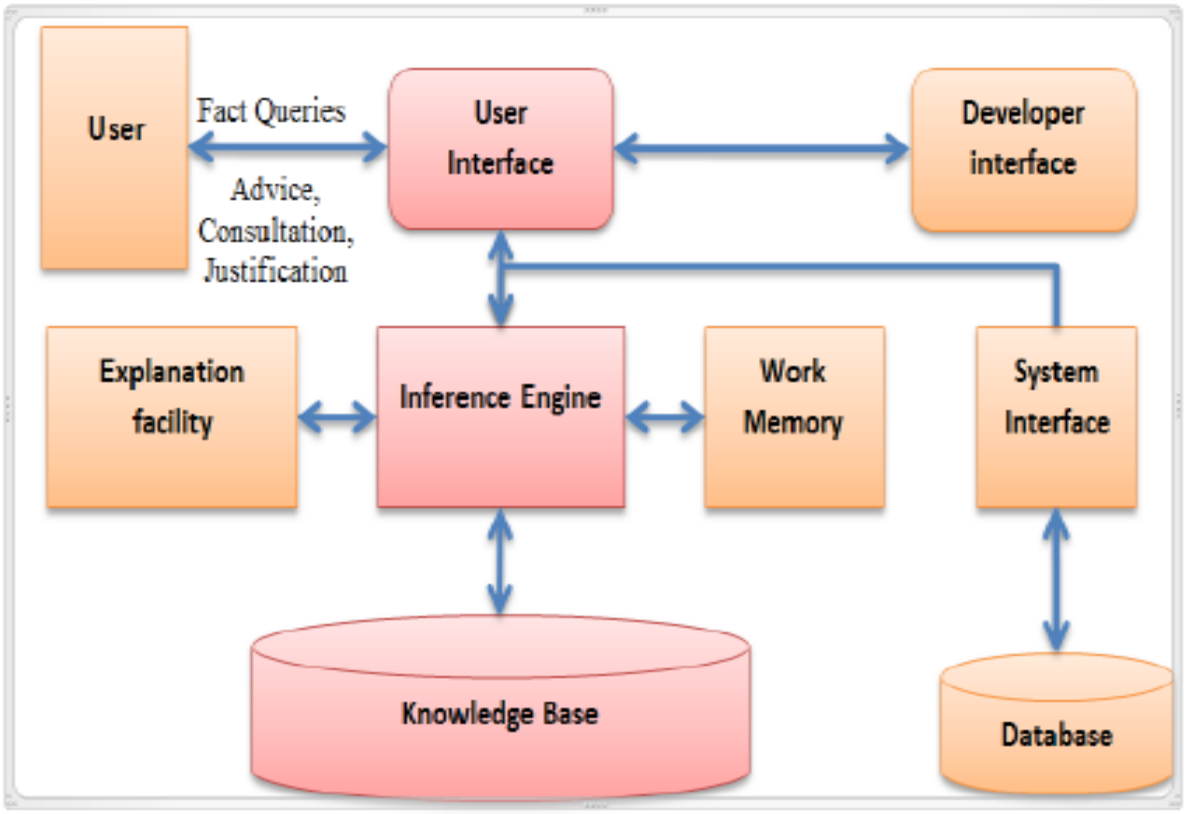

Figure 3: Architecture of an Expert system

3.1 The User Interface

- is the part of the system which takes in the user‘s query in a readable form and passes it to the inference engine then displays the results to the user. It is also a mechanism by which user and system communicate.
  − Language processor for friendly, problem-oriented communication
  − It could has menus and graphics

3.2 The Knowledge Base: is the collection of facts and rules which describe all the knowledge about the problem domain. The knowledge base contains the knowledge necessary for understanding, formulating, and solving problems

− Knowledge is the primary raw material of ES

In a rule-based expert system, the knowledge is represented as a set of rules. Each rule specifies a relation, recommendation, directive, strategy or heuristic and has the IF (condition) THEN (action) structure. When the condition part of a rule is satisfied, the rule is said to fire and the action part is executed.

3.3 The inference engine: carries out the reasoning whereby the expert system reaches a solution. It links the rules given in the knowledge base with the facts provided in the database.
- Makes inferences deciding which rules are satisfied and prioritizing.
- The *brain* of the ES
- The control structure (rule interpreter)
- Provides methodology for reasoning

An inference engine tries to derive answers from a knowledge base (chooses which facts and rules to apply when trying to solve the user‘s query.

There are two types of inferences

1- Forward chaining.
2 - Backward chain*ing*

Forward Chaining

The forward chaining inference engine takes rule, and if its conditions are true, adds its conclusion to working memory, until no more rules can be applied;
i.e.
if the conditions of the rule _if
A and B then C‘ are true,
then
C is added to working memory.

- In forward chaining the system simply tests the rules in the order they occur, therefore rule order is important.

Backward Chaining

- The backward chaining inference engines tries to prove a goal by establishing the truth of its conditions;
  - i.e.

The rule _if A and B then C‘, the backward chaining engine will try to prove C by first proving A and then proving B. Proving these conditions to be true, may well invoke further calls to the engine and so on.

3.4 The working memory

- The working memory might be used to store intermediate conclusions and any other information inferred by the system from the data.

3.5 Explanation facility

The explanation facilities enable the user to ask the expert system how a particular conclusion is reached and why a specific fact is needed. An expert system must be able to explain its reasoning and justify its advice, analysis or conclusion. It is also a part of the expert system that allows a user or decision maker to understand how the expert system arrived at certain conclusions or results.

4. 0 Development of an Expert System

The process of building an expert system is called knowledge engineering. Correspondingly, developers of expert systems are referred to as knowledge engineers.

Knowledge engineering or knowledge-based engineering

Expert system technologies are varieties of artificial intelligence (AI) approaches in which decision-making knowledge is codified and modeled, the process of designing an expert system through the AI approaches is called **knowledge-based engineering**.

The process of building an expert system is a semblance of an attempt to capture rare or important expertise and embody it in computer code. This process involves a rigor of intellectual cloning to inject an expert knowledge into an artificial object. Expert system builders, the knowledge engineers, find out from experts what they know and how they use their knowledge to solve a particular problem. Once this debriefing is done, the expert system builders incorporate the knowledge and expertise in computer programs, making the knowledge and expertise easily replicated, readily distributed, and essentially immortal

It consists of three stages:

1. **Knowledge acquisition**: the process of obtaining the knowledge from experts (by interviewing and/or observing human experts, reading specific books, e.t.c).
2. **Knowledge representation**: selecting the most appropriate structures to represent the knowledge (lists, sets, scripts, decision trees, object–attribute–value triplets, e.t.c).
3. **Knowledge validation**: testing that the knowledge of ES is correct and complete

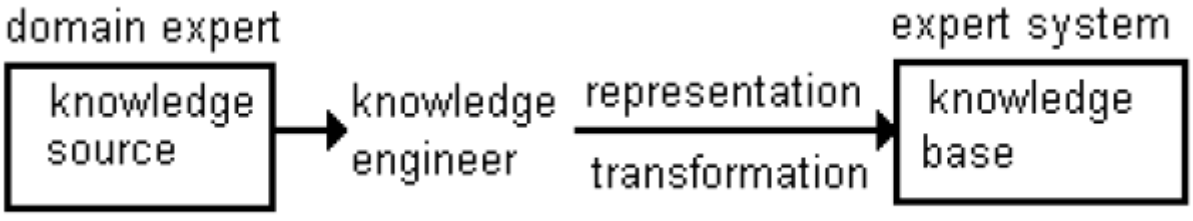

Figure 4: The structure of knowledge engineering process

The concept of knowledge-based engineering grew out of the early work on expert systems in the seventies. With the growing popularity of knowledge-based systems, there arose also a need for a systematic approach for building such systems, similar to methodologies in main-stream software engineering. Over the years, the discipline of knowledge engineering has evolved into the development of theory, methods and tools for developing knowledge-intensive applications. In other words, it provides

guidance about when and how to apply particular knowledge-presentation techniques for solving particular problems.

In this course, two related aspects of knowledge engineering will be described; these are iterative development and prototyping.

Expert systems are developed iteratively in a series of repeated steps. These steps roughly consist of knowledge acquisition, followed by system design (or modification of an existing design, system development (including knowledge entry) and system testing and evaluation.

4.1 The steps in a typical expert systems analysis and design methodology are summarized below.

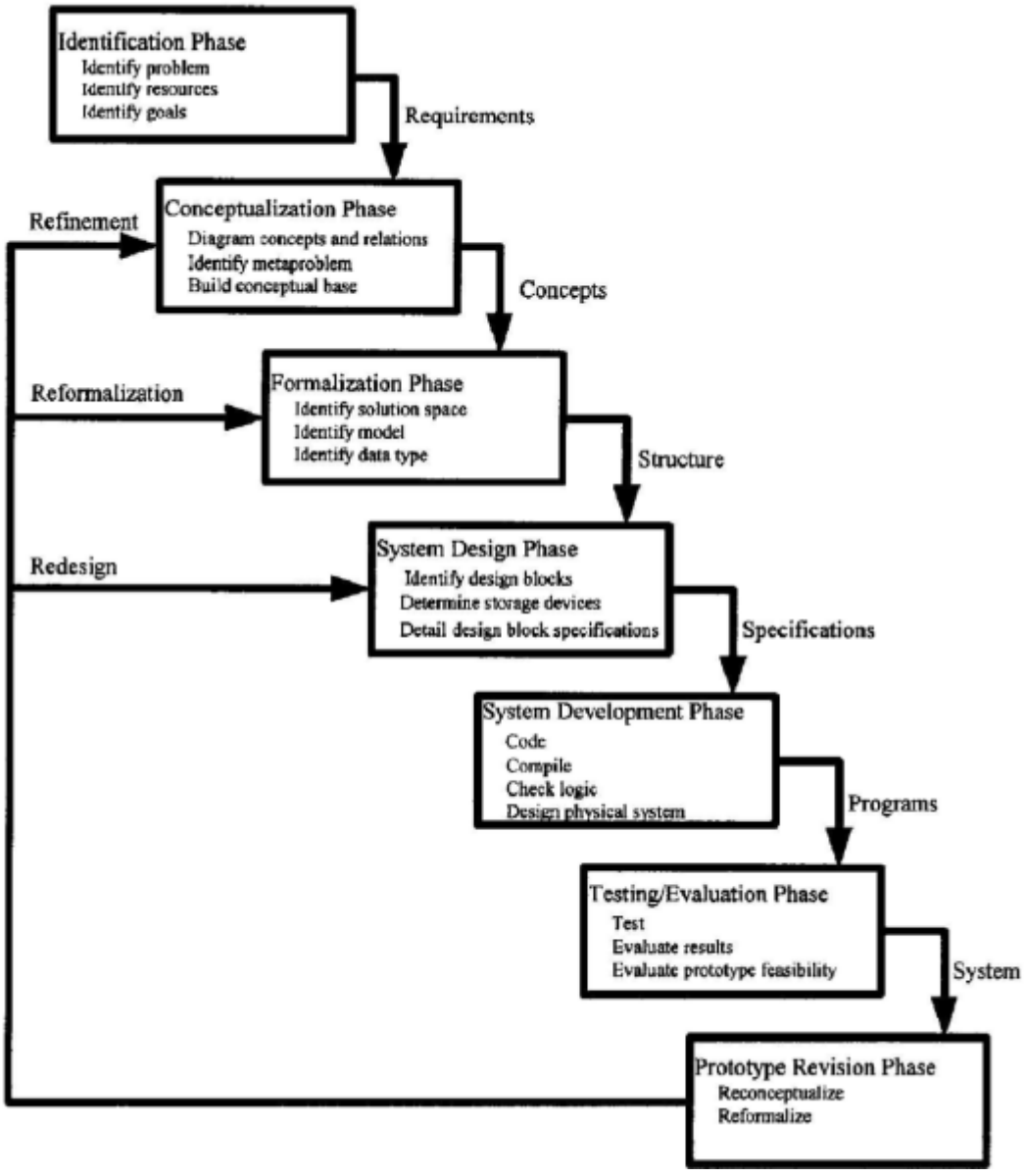

Figure 5: Iterative Development of an Expert System

1. Identification phase

The first step in the identification phase, *Identify problem,* is similar to the problem definition phase in the traditional systems development life cycle. The objective is to identify, characterize, and define the problems the system will be expected to solve and then partition the problem into appropriate sub-tasks.
Once the problem is defined, the resources necessary for acquiring knowledge, implementing the system, and testing the system are identified. Typical resources include knowledge, time, computing facilities, and money. Because expert systems are expensive and creating one takes considerable time, a feasibility study is often conducted before work progresses beyond this point.
In addition to identifying resources, the expert system analysts and/or designers also identify the system's goals and objectives. It is helpful to identify and explicitly document the goals because certain design approaches, such as heuristic search, breadth search, depth search, and reasoning are goal-driven.

2. Conceptualization phase
The central task of the conceptualization phase is to diagram the system's key concepts and relations to define a conceptual base for a prototype system. Key objectives include separating the inference engine from the problem domain, factoring (analyzing) the problem into meta-problems, identifying the system's key concepts and relations, and testing those concepts and relations by challenging them (with specific examples of problem-solving activities) to ensure that they cover every general case. Many of the tools and techniques are used in this phase.

3. Formalization Phase
The formalization phase involves mapping key concepts, sub-problems, and information flow characteristics isolated during conceptualization into more formal representations based on various knowledge engineering and problem solving tools and knowledge representation frame-works. The key objectives are to identify the solution space (a domain with a collection of all possible solutions), the hypothesis space (the hypothetical solution space), the underlying model, and the characteristics of the data. To define the structure of the hypothesis space, the systems analysts or designers must formalize the concepts (knowledge in an abstract format that can be used to guide a searching or reasoning process) and determine how they are joined to form a hypothesis.
The concepts provide clues about the nature of the space such as if it is finite, if a hierarchy must to be considered, if certain levels of abstraction can be applied, and if a specific class of the concept must be generated. Such searching techniques as blind search, heuristic search, and abstracting the solution space are often used. Reasoning techniques such as assumption building, justification building, and the constraints and goal technique help to identify the underlying model of the process used to generate solutions in the domain.

4. System design phase
During the system design phase (sometimes called the logical design phase) the analyst and/or designer specifies how the system will meet the requirements identified during the previous three phases. Typically, the reports and other outputs the systems must produce are defined first. This phase is similar to the design stage in the traditional systems development life cycle. Note, however, that the representation schemes used to describe knowledge differ from traditional methodologies.
Using the knowledge you have acquired and the tool you have selected, you can now begin the design of the expert system. First, you will need to create an outline, a hierarchal flow chart, a matrix, decision table, or other format that will help you organize and understand the knowledge. Using these aids, you will convert the knowledge in to IFTHEN rules. It is best to follow the specific procedures recommended by the software tool you are using. Once the basic design is complete, you can begin using the tool to create a prototype of one segment of the system. Translate a portion of the knowledge into rules and test the newly created segment. Test the concept before going ahead with the entire program.

5. System development phase
A prototype expert system is created during the system development (or physical design) stage. This stage is similar to the development stage in the traditional system development life cycle. Once you have satisfied yourself that the system is going to work satisfactorily, now you can begin to expand the prototype into the final system. The best way to go about this is to expand the prototype one segment at a time.

6. Testing and evaluation phase

During this phase, the prototype system is evaluated. This phase parallels the testing stage in the traditional system development life cycle. However, in addition to the testing tools and techniques, expert systems utilize a dynamic testing technique to verify the reasoning and/or inference process.

After the expert system has been developed, you will need to spend some time to testing and debugging it. No expert system will be perfect the first time, and a considerable amount of work will be required to validate it.

User feedback will show you where to make final changes, corrections, and additions to achieve the desired performance.

7. Prototype revision phase

An expert system evolves over time, calling for almost constant revision, a trait expert systems share with most prototypes. Based on the results of the testing/evaluation phase, concepts and relations are refined, the solution space, the model, the data characteristics are re-formalized, and the system is redesigned.

8. **Maintain the System**

An important part of expert system development is ongoing maintenance, updating the system with new knowledge, removing knowledge that is no longer applicable, and otherwise fine tuning the system to keep it fully current and applicable to the problem.

5.0 Expert system development Software tools

Most expert systems are developed via specialized software tools called shells. Shells provide framework to produce an expert system.so the knowledge base and rules are simply added to this frame work. These shells come equipped with an inference mechanism (backward chaining, forward chaining, or both), and require knowledge to be entered according to a specified format. They typically come with a number of other features, such as tools for writing hypertext, for constructing friendly user interfaces, for manipulating lists, strings, and objects, and for interfacing with external programs and databases. These shells qualify as languages, although certainly with a narrower range of application than most programming languages.

Picking the right tool for building the expert system is an important step. Sometimes, more than one tool will be used during the course of a project. Typically, one tool is used for prototyping, but a different tool is chosen for large-scale development and delivery.

The expert system programing and tools simplify the job of constructing an expert system. They range from very high-level programming language to low-level support facilities. These are divided into four major categories

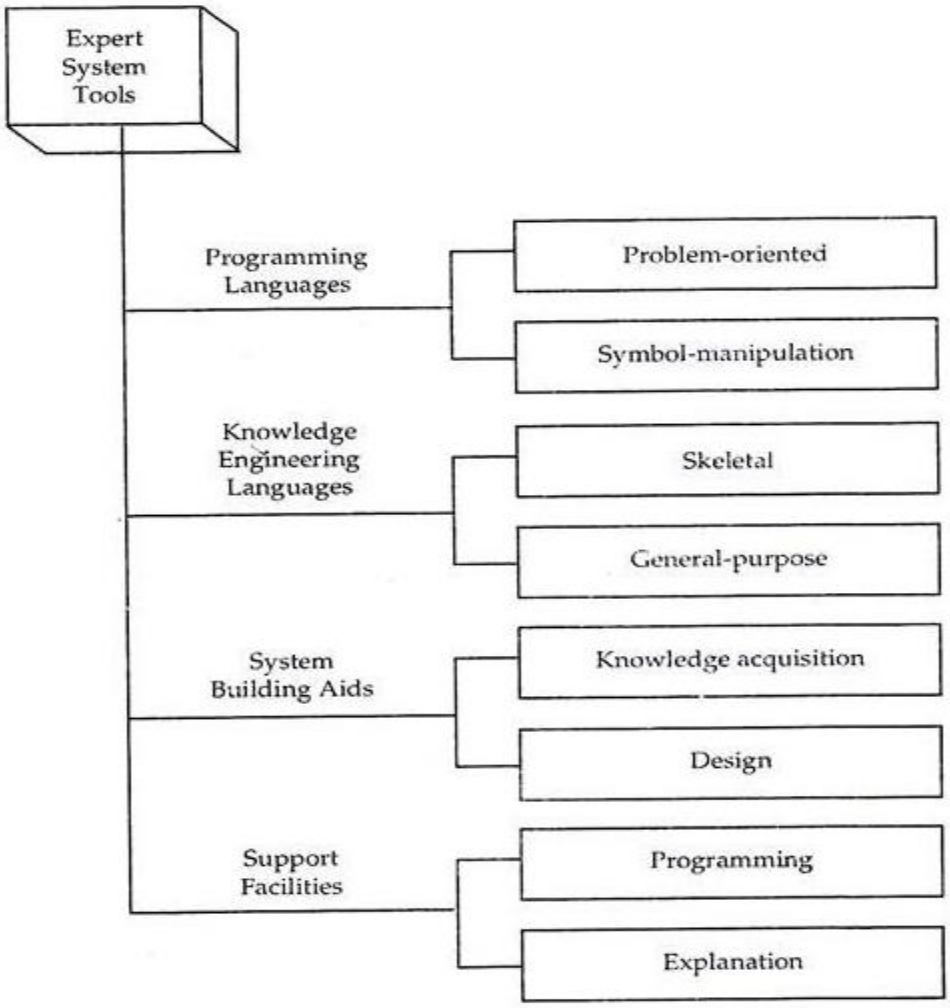

Figure 6: Shows the types of tools for expert system design

5.1 Programming languages for expert systems development

Most important programming languages used for expert system applications are generally either problem-oriented languages such as FORTRAN and PASCAL, or symbol-manipulation languages such as LISP (list programming language) and PROLOG (programming logic language). Problem oriented languages are designed for particular classes of problems; e.g., FORTRAN has convenient features for performing algebraic calculations and is most applicable to scientific, mathematical and statistical problem area.
Popular systems building products include Expert-Ease, ADVISE, RULEMASTER, SEEK, and RULE WRITER. Other relevant products include KEE, KMS, RLL, SRL, SRL+ (frame-based), APES and HSRL (logic based), ROSS,SMALLTALK, and KBS (object oriented), INTERLISP and PSL (procedure-oriented),and ARS, ART, EXPERT, EXPERT-II, OPS5, RITA, and ROSIE (rule-based).

Expert system are typically written in languages like LISP and PROLOG or even CLIPS (C language Integrated Production System Developed in mid 1980s). The use of these languages in the development of an expert system simplifies the coding process. The major advantages of these languages as compared to conventional programming languages is the simplicity of the addition, elimination or substitution of new rules and memory management capabilities. The desirable features of these languages are:
Knowledge Representation: CLIPS provides a cohesive tool for handling a wide variety of knowledge with support for three different programming paradigms.
object-oriented and procedural: The procedural programming capabilities provided by CLIPS are similar to capabilities found in conventional languages such as Python, R,C, Java, Ada, and LISP.

Portability: CLIPS is written in C for portability and speed and has been installed on many different operating systems without code changes. Operating systems on which CLIPS has been tested include Windows XP, MacOS X, and Unix. CLIPS can be ported to any system which has an ANSI compliant C or C++ compiler.
CLIPS comes with all source code which can be modified or tailored to meet a user's specific needs.
Integration/Extensibility: CLIPS can be embedded within procedural code, called as a subroutine, and integrated with languages such as C, Java, FORTRAN and ADA.
CLIPS can be easily extended by a user through the use of several well-defined protocols.
Interactive Development: The standard version of CLIPS provides an interactive, Text-oriented development environment, including debugging aids, on-line help, and an integrated editor. Interfaces providing features such as pulldown menus, integrated editors, and multiple windows have been developed for the MacOS, Windows XP, and X Window environments.
Verification/Validation: CLIPS includes a number of features to support the verification and validation of expert systems including support for modular design and partitioning of a knowledge base, static and dynamic constraint checking of slot values and function arguments, and semantic analysis of rule patterns to determine if inconsistencies could prevent a rule from firing or generate an error.
Fully Documented: CLIPS comes with extensive documentation including a Reference manual and a User guide.
Low Cost: CLIPS is maintained as public domain software.

5.2 Knowledge Engineering Languages:

A knowledge engineering language is a sophisticated tool for developing expert systems, consisting of an expert system building language integrated into an extensive support environment. A programming language is an artificial language development to acquire knowledge, accept/reject knowledge to control and direct the operation of a computer.

A knowledge engineering (KE) language is a type of programming language designed to construct and debug expert system. K.E. language provide certain faculties for building expert system; they are flexible than many programming languages with regard to how knowledge can be represented and manipulated. Knowledge engineering languages can be categorized as either skeletal systems or general-purpose systems. AI researchers developed these languages expressly for building expert systems.

A skeletal knowledge engineering language is simply a stripped down expert system-that is, an expert system with its domain-specific knowledge removed (shell), leaving only the inference engine and support facilities. The general-purpose knowledge engineering language can handle many different problem areas such as knowledge extraction, giving inference or making user interface though its use is rather tedious. These languages have a range of generality and flexibility

5.3 System-Building Aids

The system-building aids consist of programs which help acquire and represent the domain expert’s knowledge and programs which help design the expert system under construction. These programs address very difficult tasks; many are research tools just beginning to evolve into practical and useful aids, although a few are offered as full-blown commercial systems.

Compared with programming and knowledge engineering languages, relatively few system-building aids have been developed. Those which exist fall into two major categories; design aids and knowledge acquisition aids. The AGE system exemplifies design aids, while TEIRSIAS, MOLE and SALT exemplify knowledge acquisition, TIMM system construction and SEEK knowledge refinement aids.

5.4 Tool Support Environment Facilities
These are simply software packages which come with each tool to make it more user-friendly and more efficient.

Environments which can be further categorized into:

(i) Those required during the development t of the expert system programs, such as debugging aids and knowledge base editors.

(ii) Those required to enhance the capabilities of the developed programs, such as input/output facilities and explanation mechanism. Although few expert system tools support all these tools, they all support some of them. These facilities are usually available as part of K.E. language and are designed to work specially with that language.

Activity B: What is knowledge engineering?

Conclusion

At the end of this unit, you will be able to:

- Understands the concept of knowledge engineering
- Discus the components of expert system
- Explain the various steps of developing an expert system
- Identify different development tools for expert systems

Summary

- In this section we have exposed the concept of knowledge engineering; the components of expert system, these components are user interface, inference engine, knowledge base, working memory, explanation facility. Lastly, we discuss the various steps in developing an expert systems and the development tools

Tutor Marked Assignment

- What do you understand by the term knowledge engineering
- With the aid a well labeled diagram, discus the various step of developing an expert system.
- Discus the components of an expert system
- Explain the various expert system development tools

Further Reading and Other Resources

- Wagner, W. (2017). Trends in expert system development: A longitudinal content analysis of over thirty years of expert system case studies. Expert systems with applications. doi: 10.1016/j.eswa.2017.01.028
- Vizureanu, P. (2010). Expert Systems. Intech
- Watkin, J.L (2017). The CLIPS Programming Language for Building Expert Systems Emerging Languages, Spring 2017, University of Dayton. Link: https://www.youtube.com/watch?v=XX8Fxze6Np8

Unit 3: The need for Expert Systems and Applications

1.0 Introduction

This unit discusses the operation of expert system and the various application of expert system

2.0 Objectives

By the end of this unit, you will be able to:

- Discus the characteristics of expert system
- Discus the application of expert system

3.0 The Need for Expert Systems

In many organizations, problem-solving expertise is scarce. Training people to become proficient in solving specialized problems takes time and requires substantial investment. Hence, experts are always in short supply. In an operational setting, the frequency with which problems occur often exceeds the capabilities of a limited number of experts. In some situations, experts may be geographically distant from the site of the problem, or problems may occur during times experts are unavailable. Consequently, problems must be handled by less qualified personnel. This can cause delays and lead to inconsistent or uneven decision making. One example is a factory floor in which highly specialized equipment problems must be diagnosed and repairs effected. Normally, this task is performed by a trained expert having years of experience. Work proceeds normally if the frequency of malfunctions is relatively low and the expert is readily available. However, a company may have factories at different locations, or the factories may operate around-the-clock. Under these circumstances, demand for experts may quickly exceed their availability, resulting in delays and problems. One solution is to develop an expert system to help identify frequently occurring machine malfunctions and suggest solutions. Such an expert system could be deployed on the factory floor and used to solve routine problems that would otherwise have to be handled by the expert. If the expert system could solve the problems the expert normally solves, delays could be eliminated and productivity improved. Copies of the expert system could be distributed throughout the company, making expertise available at several locations around the clock.

This is a simplified description of the productive use of an expert system. Certainly, there are potential problems in this scenario that could defeat effective use of expert system technology.

4.0 When is an Expert System Appropriate?

Here are factors which suggest an expert system is appropriate.

• Need justifies cost and effort
• Human expertise not always available
• Problem requires symbolic reasoning
• Problem domain is well structured
• Traditional computing methods fail
• Cooperative and articulate experts exist
• Problem is not too large

5.0 Expert System Application Areas

Expert systems have gained wider applications in different areas of human endeavor, especially in roles where human expertize are need. Some of these areas where expert systems are applied are discussed below:

1. Accounting & Finance

The financial services industry has been a vigorous user of expert system techniques. Advisory programs have been created to assist bankers in determining whether to make loans to businesses and individuals. Insurance companies have used expert systems to assess the risk presented by the customer and to determine a price for the insurance. A typical application in the financial markets is in foreign exchange trading.

Other areas in accounting and finance applications are - Cost Code Selector, Stock & Commodity Trading, Portfolio Construction, Home Purchasing, Financial Planner Training and Selection, Personal Tax Advisor, Detecting Insider Trading, Organizational Services, Credit Analysis Advisor, Bank Loan Identification, Credit Control, Loan Documentation, Assessment of Risk and Fraud in Financial Institutions, Aid in Tax Form Completion, Commercial Loan Approval Predictor...

2. **Agricultural -** Irrigation and Pest Control, Crop Variety Selection and Management, Soil Characterization and Utilization for Specific Areas, Fertilizer, Climate and Soil Interaction and Analysis, Salmon Stocking Rates and Species Selection, Forest Inventory, Weed Identification, Soil Conservation, Tree Selection Based on Environmental Conditions, Planning and Design of Agro forestry Systems.

3. **Business -** Alternatives for Fragmented Industry, Advertising Copy Development, Shipping Documentation and Routes, Market Advisor for Process Control Systems, Demographic and Market Assessment, Product Performance Trouble-shooting, Sales Personnel Assessment, Account Marketing, Invention Patent Ability, Salary & Benefit Planning, Client Profile Business Application Selection, Professional Service Selection, Career Goal Planning, Pension fund Calculator, Unemployment Insurance Eligibility.

4. **Chemical -** Hazard Evaluation, Chemical Facilities Procedures, Correct Propellant Ingredient Mixture, Pollution Control Technology Permits, Common Metal and Alloy Identification, Real-time Process Controlled City Waste Water Management, Solvent Selection for Chemical Compounds, Pottery Glaze Recipe and Identification, Pulp Bleaching Advisor, Toxicity of Laboratory Chemicals, Lime Recognition System, Process Diagnosing and Troubleshooting...

5. **Computer -** Software System Diagnostic Modeling, Application Sizing, Software Quality Assurance, Program Classification, Locating Component Failure & Analysis, New Technology Selection, Training Systems, Custom Hardware Diagnostics, Decision Support Systems, MIS Support System, Program Library Maintenance, Fault Detection and Diagnostics of Wide Area Networks, Analysis of Statistical Data, Personal Computer Configuration, Shielding Technique Selection, Database Design, Computerized Technical Service Representation, Hardware and Software Selection by Non-Technical users, New User of Computer Assistance, Documentation Recommendations to Users, Monitoring, Repair Assistance and Problem Prediction of Operating System.

6. **Construction -** Pavement Rehabilitation & Design, Structural Damage Assessment, Equipment Evaluation & Selection, Material Costing & Selection, Project Scheduling, Cost Estimating, Evaluating Multifamily Housing Projects, Work Zone Safety Trainer/Advisor, Weld Procedure Selection and Cost Estimating, Soil Compacting, Fire Code Advisor, Alarm Management System.

7. **Education -** Library Reference Material Recommendation, Interpretation of Statistical Quality Control Data, Teaching Mineral, Rock and Fossil Identification, Student Financial Aid Eligibility, Analysis of Metal Cautions, Fire Department Emergency Management Advisor, Medical Student Diagnostic Systems, Dentistry Advisor, Telephone Customer Support Instruction, Gas Turbine Training, Industrial Training, Patient Care Advisor for Student Nurses.

8. Engineering

Expert systems are widely used in engineering, they can help in configuration, whereby a solution to a problem is synthesized from a given set of elements related by a set of constraints. The technique has found its way into use in many different engineering industries, for example, modular home building, manufacturing, and other problems involving complex engineering design and manufacturing. Other areas of ES applications in engineering are:

- Engineering Change Control Demonstrating, Diesel Engine Lube Oil Wear Analysis, Super-alloys Phase Analysis, Equipment Diagnostics, Electronic Semiconductor Failure Testing, Parts List Selection & Sizing, Power Generation System Scheduling, Component Failure Prediction, Machining Advisor, Numerically Controlled Machine Tool Selection, Petrochemical Plant Process Control, Control Panel Layout Design, Material and Process Design Advisor...

9. **Insurance -** Rating for Substandard Life Insurance, Workers Comp Classification, Underwriting Assistance, Social Security Help Desk and Benefit Identification, Unemployment Insurance Eligibility.

10. Medical: Expert systems have strong presence in medical applications. It should be noted that Medical diagnosis was one of the first knowledge areas to which ES technology was applied. Other medical applications of expert systems are: Admission Protocols, X-Ray Analysis, Hematological Diagnoses, Psychiatric Interviewing, Pediatric Auditory Brainstem Response Interpretation, Medical Decision Making, Respirator Selection for Preschool Children, Health Services Utilization and modeling, Diagnostic Systems, In-Vitro Fertilization, Symptom Analysis, Voice-Driven Lab Diagnosis, Rehabilitation Feasibility Strategies, Billing and Account Management, Disease Research. E.t.c

11. **Military**, Government & Space Related - Submarine Approach Officer Training, Combat Methodology Selection, Radar Mode Workstation Designing, Federal Contract Management, Severe Weather Forecasting, Shuttle Payload On-Orbit Analysis, Metals Materials Selector, GB Satellite Abnormality Correction, Thermal Analyst, Selection of Non-materials in Aerospace Applications...

12. Trouble-Shooting

This class comprises systems that deduce faults and suggest corrective actions for a malfunctioning device or process. This has wide applications in areas such as

- Airplane Starting Systems, Data Communications, Test and Repair of a PCB, Gas Turbine Control System, Bearing System Failures, Telecommunications Difficulties, Real-Time Process Control, Meterman's Assistant System, Mechanical Equipment and Systems Diagnosis, Weld Flaw Detection, Web Break Diagnosis in Paper Milling, Power Plant Turbine Generator Bearing Maintenance System.

13. **Knowledge Publishing:** This is a relatively new, but also potentially explosive area. The primary function of the expert system is to deliver knowledge that is relevant to the user's problem, in the context of the user's problem. The two most widely distributed expert systems in the world are in this category. The first is an advisor which counsels a user on appropriate grammatical usage in a text. The second is a tax advisor that accompanies a tax preparation program and advises the user on tax strategy, tactics, and individual tax policy.

Many, Many Others - Telephone System Configurator, Training Material & Product Selection, Manufacturing Resource Planning, Production Scheduling, Service Networking, Airline Scheduling, Cost/Benefit Analysis, Planning Implementation, Career Development, Quick Proposal Estimating, Credit Control, Product Development, Telephone Call Screening, Real Estate Market & Mortgage Credit Analysis, Retirement Planning, Sales Analysis...

The spectrum of applications of expert systems technology to industrial and commercial problems is so wide as to defy easy characterization. The applications find their way into most areas of knowledge work. They are as varied as helping salespersons sell modular factory-built homes to helping NASA plan the maintenance of a space shuttle in preparation for its next flight.

Activity C: State the application areas of expert system

6.0 Conclusion

At the end of this unit, you have been able to learn:

- The various applications of expert system
- When is an expert system desirable?

7.0 Summary

Expert system is found applicable to many arrears of human endeavours: Medicine, Agiculture,. Banking e.t.c. Expert system is invented to argument the scarcity of expert in a particular domain.

Tutor Marked Assignment

- Discus the various application of expert system
- Discuss when an expert system is desirable in an organization.

Further Reading and Other Resources

Christopher, et al (1991). Guide to Expert System Building Tools for Microcomputers

Unit 4: Knowledge Representation in expert systems

1.0 Introduction

In this unit, you will learn about how knowledge is represented in expert systems.

2.0 Objectives

By the end of this unit, you should able to:

- Discuss knowledge representation in expert systems
- Explain the different types of knowledge representations in expert systems
- Discus the benefits and disadvantages of each types of knowledge representations

3.0 Definition: Knowledge-representation is the field of in artificial intelligence that focuses on capturing information about the world that can be used to solve complex problems in a particular domain such as diagnosing a medical condition.

Knowledge and Representation are two distinct entities. They play central but distinguishable roles in intelligent system design.

Knowledge is a description of the world. It determines a system's competence by what it knows. On the other hand
Representation is the way knowledge is encoded so that the computer can understand on how to solve a problem. It defines a system's performance in doing something

Knowledge is a **progression** that starts with data which is of limited utility.
1. **Data** is viewed as collection of disconnected facts
Example: It is raining
2. By organizing or analyzing the data, we understand what the data means, and this becomes **information**. It provides answers to "who","what", "where", and "when".
Example: The temperature dropped 15 degrees and then it started raining
3. The interpretation or evaluation of information yield **knowledge.** It provides answers as "how".
Example: If the humidity is very high and the temperature drops substantially, then atmospheres is unlikely to hold the moisture, so it rains
4. An understanding of the principles embodied within the knowledge is **wisdom**. It provides answers as "why".

- Knowledge is the most abstract and exists in the smallest quantity. Knowledge itself can have levels of abstraction: concrete (knowledge about the specific problem), domain specific (class of problems) and
abstract (many classes of problems).

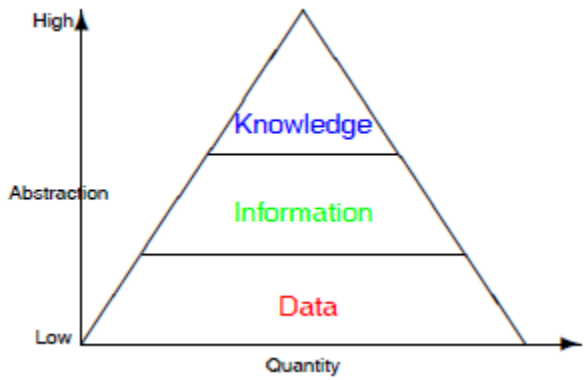

Figure 7: Showing the level of abstraction of knowledge from data

3.1 Why do we need knowledge Representation?
Knowledge representation is crucial because problem solving requires large amount of knowledge and some they should be mechanism for manipulating that knowledge into computer understandable form.

How do we represent what we know?
Representing knowledge requires an analysis to distinguish between
knowledge “how” and knowledge “that.
Knowing " how to do something".
e.g. "how to drive a car" is a **Procedural knowledge**
Knowing "that something is true or false".
e.g. "that is the speed limit for a car on a motorway" is a **Declarative knowledge**.

3.2 Knowledge is categorized into two major types: Tacit and Explicit
The term“Tacit “corresponds to "informal" or "implicit" type of knowledge that is not codified.
1. Exists within a human being; it is embodied.
2. Difficult to articulate formally
3. Difficult to communicate or share
4. Hard to steal or copy.
5. Drawn from experience, action, subjective insight.

On the other hand the term “Explicit” corresponds to "formal" type of knowledge that is codified.
1. Exists outside a human being; it is embedded.
2. Can be articulated formally
3. Can be shared, copied, processed and stored
4. Easy to steal or copy
5. Drawn from artifact of some type as principle, procedure, process, concepts

Knowledge Typology Map
It shows the relationship between – Tacit and Explicit knowledge.
Tacit knowledge comes from "experience", "action", "subjective", "insight".
Explicit knowledge comes from "principle", procedure", "process", "concepts".
Facts: are data or instance that is specific and unique.
Concepts: are classes of items, words, or ideas that are known by a common name and share common features.

Processes: are flows of events or activities that describe how things work rather than how to do things. Procedures: are series of step-by-step actions and decisions that result in the achievement of a task.
Principles: are guidelines, rules, and parameters that govern; principles allow to make predictions and draw implications;

- ✓ A good knowledge representation enables fast and accurate access to knowledge and understanding of the content

3.3 Knowledge Representation Methods

Knowledge is represented by three methods in expert system; these are:

I. Production Rules
II. Semantic Net
III. Frames and Logic

Production Rules or Production System:
Rules are used to represent relationships. Rule-based knowledge representation employs
IF **condition** (premise or consequent)
THEN **action** (goal or antecedent) statements.
For example,
IF the heating element glows **AND** the bread is always dark
THEN the toaster thermostat is broken
When the problem situation matches the IF part of a rule, the action specified by the THEN part of the rule is performed

- Production rules are one of the most popular and widely used knowledge representation languages
- Production rule system consists of three components

I. Working memory contains the information that the system has gained about the problem thus far.
II. Rule base contains information that applies to all the problems that the system may be asked to solve.
III. Interpreter solves the control problem, i.e., decide which rule to execute on each selection - execute cycle.

Advantages of Production System rule method:

- ✓ Naturalness of expression
- ✓ Modularity
- ✓ Restricted syntax
- ✓ Ability to Represent Uncertain Knowledge

Disadvantages of Production System:

- ✓ Inefficient
- ✓ Less expressive
- ✓

Semantic Net
It is **formalism/mechanism** for representing information /Knowledge about objects, people, concepts and specific relationship between them.

The syntax of semantic net is simple. It is a network of labeled nodes and links.
It is a directed graph with nodes corresponding to concepts, facts, objects etc. and arcs showing relation or association between two concepts.

The commonly used links in semantic net are of the following types.
isa → subclass of entity (e.g., child hospital is subclass of hospital)
inst → particular instance of a class (e.g., India is an instance of country)
prop → property link (e.g., property of dog is 'bark)

Representation of Knowledge in Semantic Net
Every human, animal and bird is living things that breathe and eat. All birds can fly.
All man and woman are humans who have two legs. Cat is an animal and has a fur.
All animals have skin and can move. Giraffe is an animal who is tall and has long legs.
Parrot is a bird and is green in color

Inheritance in Semantic Net
-Inheritance mechanism allows knowledge to be stored at the highest possible level of abstraction which reduces the size of knowledge base.

- It facilitates inference of information associated with semantic nets.
- It is a natural tool for representing taxonomically structured information and ensures that all the members and sub-concepts of a concept share common properties.
- It also helps us to maintain the consistency of the knowledge base by adding new concepts and members of existing ones.

- Properties attached to a particular object (class) are to be inherited by all subclasses and members of that class.

Advantages of Semantic nets

- Easy to visualize
- Formal definitions of semantic networks have been developed
- Related knowledge is easily clustered.
- Efficient in space requirements
- Objects represented only once
- Relationships handled by pointers

Disadvantages of Semantic nets

- Inheritance (particularly from multiple sources and when exceptions in inheritance are wanted) can cause problems.
- Facts placed inappropriately cause problems.
- No standards about node and arc values

Frame

- Frame is a semantic net with properties
- It represents general concept or specific entry
- Frames represent objects as sets of slot/filler pairs
- Object can contain programs as well as data (if-needed, if-added, if-removed).
- The utility of frames lies in hierarchical frame system and inheritance.
- This makes it easy to construct and manipulate a complex knowledge base.
- Frames are implicitly associated with one another because value of a slot can be another frame

There are three components of a frame
(i). Frame name
(ii). Attributes (slots)

(iii). Values (Fillers)
- Fillers can be links to other frames

Advantages
− Domain knowledge model reflected directly
− Support default reasoning
− Efficient
− Support procedural knowledge
Disadvantages
- Lack of semantics
− Expressive limitations

4.0 Overview of Natural Language interface for Expert Systems
A Natural language Interface (NLI) is defined as an interface that has the ability to interact with users using human language such as English, as opposed to computer language, a command line interface, or a graphical user interface. The interface takes as input either written text or spoken speech. These types of input may be utilized individually or in a combination to produce a multimodal input interface. NLIs are usually only capable of understanding a restricted subset of a human language (usually restricted to a certain domain) and generate more or less pre-packaged responses. The user would have to learn to utilize a small subset of the English language in order to operate most of these interfaces. Some experts however feel that this is not an effective NLI as users would have to learn how to use the system before being able to utilize it effectively. They argue that applications that utilize natural language should stimulate conversation between the human user and the computer in order to have a successful communication between the two parties. Human-to-human conversations take place by taking turns between speaker and listener. When NLIs are designed this should therefore be taken into account. NLIs were first utilized for human-computer interaction through natural language in 1966 with a system, called ELIZA, created by Joseph Weizenbaum. ELIZA was a simple conversational agent that was capable of parsing simple sentences and utilizing them to pick out keywords. These keywords were then utilized for substitutions to turned into questions. ELIZA was not capable of holding a long conversation with a user as most statements made by the user were merely turned into questions. Though ELIZA did not utilize any sophisticated processing techniques, it was still a notable NLI as it was the first natural language interface.

Natural language can be processed by computers through *speech recognition, speech synthesis, text-based pattern matching* or *gesture interaction.*

Speech recognition has come of age in recent years but has not matured to an extent that it can be utilized to have a conversation with a computer. Most speech recognition systems are restricted to certain keywords within a domain and, therefore cannot be utilized for accomplishing tasks outside a particular domain (Dusan and Flanagan 2004). Furthermore, these systems do not adequately cover all utterances a user might utilize in order to accomplish a task in a certain domain. Research in the area of natural language processing is moving towards the creation of systems that are capable of learning human knowledge and obtaining related knowledge as a conversation proceeds (Dusan *et al*. 2004).

Speech synthesis is the process of outputting simulated human speech. Speech

synthesis has been more successful than speech recognition and has a variety of applications. Contact centres use speech synthesis to present the customer with menus with which they interact. Furthermore feedback is also provided to the customer using speech synthesis.

Text-based pattern matching has mostly been utilized by conversational agents or search engines. These systems utilize pattern matching in order to interact with the user. An example of text-based pattern matching is called ALICE, which utilizes pattern matching techniques to converse with the user. Pattern matching is the act of checking text for a given structure and if it matches a certain task is performed. In the case of ALICE when a pattern is matched a template will be triggered. *Gesture interaction* is the process of analyzing human gestures. Humans usually gesture when using human language in order to interact with other human beings. These gestures can convey the mood and sometimes the context of a conversation. In some cultures bowing in front of someone is a method of saying hello (therefore the context of the conversation at that point is introduction). Gesture interaction is usually utilized as a complimentary technique to speech. They provide a method through which context can be maintained in a conversation. Furthermore, the mood of the conversation can also be understood by the system.

4.1 Conversational Speech Interfaces

A conversational speech interface is one that utilizes speech as an interaction technique and is an example of a natural language interface. This interface operates utilize the following functions: speech recognition (input) and speech synthesis (output). A major benefit of incorporating speech in applications is that it comes naturally to humans. Most people find speaking and listening easy. Though speech is easy for humans it is not so easy for computers. The reason for this is that speech technologies lack 100% accuracy (Lai and Yankelovich 2003), and there is a problem with ambiguity when utilising natural languages. The main problem when designing these interfaces is that system developers do not design these systems with speech in mind from the beginning but rather in terms of the graphical user interface. This brings about a problem of merely designing a command line interface which utilizes speech through a graphical user interface (Lai *et al*. 2003). Another crucial factor in determining whether or not the application will be successful is to determine whether there is a clear benefit to utilizing speech. This involves assessing whether speech is absolutely necessary, in other words when the users hand or eyes are busy or when the task to be completed cannot be accomplished without the use of speech. Speech is not suitable in situations such as when large amounts of information need to be presented to the user.

4.2 Speech Recognition

Speech recognition can be seen as the process by which a computer can identify the parts of human speech. The process starts with the user uttering something into a microphone and ends with the computer accomplishing a task. The solution that has not been successfully implemented by humans for computers is to accurately identify all possible words spoken by any person in any environment.

Systems performance in speech recognition can be affected by a number of factors including large vocabularies, multiple users, continuous speech and noisy environments. When a user speaks into a microphone, phonemes are extracted from the user speech. Phonemes are the linguistic units of human language (Zue, Cole and Ward 2000; Matthews 2002). These sounds are grouped together to form words

utilized in human language. When the sounds are grouped together the actual process of understanding user input is initiated. Over the years, many approaches have been utilized but only two will be discussed here namely pattern matching and knowledge-based approaches. These two approaches are not mutually exclusive and are discussed below:

- Pattern matching - The goal of pattern matching is to take an unknown pattern and compare it to a set of known and stored patterns (also known as templates). These are established through training data. Templates are utilized to compare the pattern and compute a similarity score (Zue et al. 2000; Schroeder 2004). The template with the highest score is then chosen as it will have the highest acoustic similarity to the users input.
- Knowledge-based approaches - This approach makes use of a rule-based expert system which utilizes a base classification of rules in order to function. However to utilize this model, a large set of rules would be needed in order to capture a great variability in speech. Rules are formed from knowledge about speech signals. This approach could be useful, however, it does not perform effectively if there is insufficient knowledge.

4.3 Speech Synthesis

Speech synthesis enables computers and other electronic devices to output simulated human speech. A computer system that is utilized for the purpose of producing human speech is known as a speech synthesizer. A text-to-speech (TTS) system is an example of a speech synthesizer that converts normal text into speech. The quality and effectiveness of speech produced by these system are measured by utilizing these characteristics:

- Base-level achievement of speech that is intelligible (the ease with which the output is understood) to humans;
- produce speech that is as natural as that of human beings in other words how natural the speech is;
- produce speech that is personalized to a particular user, in other words it has the same intonation as a person's speech; and
- the final level and highest level of achievement is to produce speech based on a person's own voice recordings so that the speech sounds as if it belongs to that person.

For a text-to-speech system to function, a narrator is utilized to record a series of text (such as reading from an encyclopedia, poetry, political news and various other texts). These narrations are carefully picked in order to ensure that every possible sound in a given language is recorded. These narrations are then sliced into the different phonemes found in the particular language and stored in a database. When the database is created, the various recorded utterances are segmented into one of the following: diaphones (sound-to-sound transitions), syllables (units of organization for a sequence of speech sounds), morphemes (smallest linguistic units that have semantic meaning), words, phrases and sentences.

The above-mentioned process occurs in the *back end* of the TTS system. The *front end* has two major tasks namely: *normalization* and *text-to-phoneme conversion*. *Normalization* is the process that occurs first, where the conversion of raw text occurs. All abbreviations and symbols are expanded into their respective words. These are

then passed down to the component that starts the *text-to - phoneme conversion*. Once the text is passed on this component makes sure that the sentence is syntactically correct. The various phonemes are then extracted for that particular sentence and produced as human speech through a speaker or any other device capable of producing sound. There are two types of speech synthesis that are utilized for commercial applications namely concatenated synthesis and formant synthesis (Karat *et al*. 2003). Concatenated synthesis employs computers to assemble narrators recorded voices into speech signals. Though this sounds fairly simple, it is very database intensive and has large storage needs to store all recorded speech. Formant synthesis, on the other hand, utilizes a rule-based expert system. This expert system applies a set of phonological rules to a specified audio waveform which simulates speech.

4.4 Text-Based Natural Language Interfaces

Text-based NLIs utilize text instead of speech in order to function. Text may be in the form of a query, sentence or a list of keywords (as used by search engines). A user will type in a question in an appropriate field (usually a textbox) and the system will retrieve information in accordance to the users query. Text-based natural language interfaces have been utilized in many applications including: databases (query and report generation), conversational agents and search engines (matching user requests to keywords). Conversational agents are a communication technology that utilizes natural language and various linguistic methods to interact with human users through natural language. Conversational agents need to satisfy two sets of requirements for them to be effective (Lester *et al.* 2004):

- they must have good language processing capabilities such that they have the ability to engage in productive conversations with the user. This involves understanding user input and employing effective dialogue management techniques; and
- they must be scalable and reliable and allow for smooth integration within business processes.

Conversational agents are generally deployed on retail websites and are utilised by customers to enquire about products and services. However as these agents gained popularity they were deployed in a variety of other domains such as education, banking, travelling agencies and a variety of others. Conversational agents have also been utilized in the virtual world and because these worlds are mostly text based in nature, they make an ideal environment for communication with the user. These types of virtual conversational agents can be found on social websites such as Second life. Another conversational agent known as ALICE utilizes simple pattern-matching techniques to engage users in a conversation. ALICE utilizes Artificial Mark-up Language (AIML), which is a derivative of XML to store its patterns and templates.

A Virtual World Personal Assistant Pattern matching is not the only technique utilized; various other methods such as Bayesian networks are also used. There are mainly two methods utilized by text-based interfaces to comprehend user input namely:

- *Pattern matching* – conversations and their responses are stored in pairs. The pattern is the user input or stimulus which is matched to produce a template which is the output of the system. The pattern can be seen as a simple text string that has to match the input exactly. The template is the output that a user receives as a response (as it was entered by the person who created the conversational agent). The biggest flaw with pattern matching is that it is difficult to maintain context of a conversation. Moreover it is impossible to cater for all possible inputs.
- *Bayesian networks* - is a graphical model that represents a set of variables and their probabilistic dependencies. It is implemented in conversational agents to maintain context in a conversation. Context can be maintained through Bayesian networks as the probability of the next question can be calculated. This probability is calculated by analyzing current and previous user input. The question with the highest probability is usually always chosen.

4.5 Shortcomings of Natural Language Interfaces

Human-computer interaction experts believe that NLIs are not as attractive as they initially appeared to be. Early literature in this field focuses on the shortcomings of these interfaces in terms of user task completion. The shortcomings are brought upon by the ambiguous nature of a natural language and heavy dependence on a huge repository of knowledge. Consider the following quote: *At last, a computer that understands you like your mother*. A computer can interpret this quote in a number of different ways (ambiguity) and thus will show us the difficulties in analyzing human language. If we look at the sentence carefully there are three interpretations that are possible:

- the computer understands you as well as you mother understands you;
- the computer understands that you like your mother; and
- the computer understands you as well as it understands your mother.

This shows us how a simple sentence can be ambiguous for a computer, while we as humans can easily rule out all other alternatives except the first one. We do so, based on a great deal of background knowledge, including understanding what advertisements typically try to convince us of (Lee 2004). There are various techniques that could be used to get around this flaw. Another technique that could be used is the *use of certain linguistic theories*, which outline certain rules that should be followed while conversing. Examples of such rules include:

- users should not give more information than necessary;
- users should not make unnecessary speech contributions than is necessary; and
- users should not intentionally make ambiguous references but rather use references that they believe will unambiguously describe exactly what they what want to achieve.

The most popular technique used to combat ambiguity is to *engage in some form of clarification or confirmation dialogue* to confirm if the interpretation is, in fact, the correct one.

However, the advances in the field of natural language processing are changing the ways languages are processed in expert systems today. A lot of these shortcomings have been overcome.

4.6 Benefits of Natural Language Interfaces

Though NLIs have their disadvantages, they also have a variety of advantages. The main reason why NLIs are seen as beneficial is because no prior training is required in order to utilize them as people use natural language in order to interact with each other on a daily basis. Other benefits of conversational speech interfaces include:

- Offers natural communication;
- Allows for physical mobility when troubleshooting problems that are not near the interface; and
- Allows for flattening of deep menu structures found in some computer systems. This would be possible as users could state their queries in natural language and would eliminate the need for menus. This would be beneficial for systems such as IVR.

Activity D: What is natural language interface?

4.0 Conclusion

This unit considers the concepts of natural language, the terms such speech synthesis, text-based pattern matching or gesture interaction as they relate to natural language interface. It considers the benefits and the shortcomings of natural interface.

5.0 Summary

In this unit, you have learnt that:

- Knowledge representation
- Different knowledge representations method
- A Natural language Interface(NLI) is an interface that has the ability to interact with users using human language such as English, as opposed to computer language, a command line interface, or a graphical user interface
- The shortcomings are brought upon by the ambiguous nature of a natural language and heavy dependence on a huge repository of knowledge.
- Natural language interface offers the following advantages:

- Offers natural communication;

Allows for physical mobility when troubleshooting problems that are not near the interface; and

- Allows for flattening of deep menu structures found in some computer systems. This would be possible as users could state their queries in natural language and would eliminate the need for menus.

6.0 Tutor Marked Assignment

- What is knowledge representation?
- List and explain different methods of knowledge representations
- What is a natural language interface
- State the its advantages and shortcomings
- Discus the terms *recognition*, speech synthesis, text-based pattern matching or gesture

interaction. and Speech recognition

7.0 Further Reading and Other Resources

MODULE 2: Classes of Expert System

1.0 Introduction

According to the working principle, the expert system can be sorted into four classes: the Rule-Based Expert System, the Frame-Based Expert System, the Fuzzy Logic-Based Expert System and the Expert System Based on Neural Network. This section will briefly introduce the model structure of these four expert systems.

2.0 Objectives

By the end of this unit, you should be able to:

- Understand the different classes of expert systems
- Explain the types of expert system
- Trace the history of Mycin
- Identify the components Mycin and trace its history

Unit 1: The rule-based expert system

1.0 Introduction

This unit explains is class of expert systems called rule-based system

2. Objectives

At the end of this unit, you should be able to:

- Explain rule-based systems
- Example of rule-based systems
- Advantages and disadvantages of rule based system

3. **The Rule-Based Expert System**: is the earliest type of expert systems and the most common way for researchers to build expert system. The expert system, which is completed by production rules, has been integrated into various research areas to help researchers solve all kinds of problems with pre-input knowledge.

A rule-based expert system is an expert system which works as a production system in which rules encode expert knowledge. Most expert systems are rule-based. It is an expert system based on a set of rules that a human expert would follow in diagnosing a problem. A classic example of a rule-based system is the domain-specific expert system that uses rules to make deductions or choices. For example, an expert system might help a doctor choose the correct diagnosis based on a cluster of symptoms, or select tactical moves to play a game.

The structure of the expert system is usually composed of three parts, including production rule, database and control strategy. Production rule is a "conditional + result" structure of the statement. For example, the condition: "if a species is a dinosaur", the result: "then the species is a reptile". If the condition in the statement is met, the result of the rule can be executed. The database is responsible for storing the conditions and results in the production rule statement. When the production rule is executed, the corresponding condition is called from the database and the rule result is put into the database as a condition for other rules. The role of the control strategy is to explain how to apply the rules, that is, in the process of solving the problem to select the appropriate rules. The

process from selecting rules to performing operations is usually the case that a problem contains several conditions, looking for matching conditions from the database, and finding the rules for the corresponding conditions. When more than one condition is matched, you need to sort the strategy to decide which rule to use first. After the rule is selected, the operation (result) section of the rule is executed. The operation of the control strategy is usually performed in a module ("inference engine"). The inference process of the inference engine can be divided into both forward and reverse.

The Rule-based Expert System has many advantages. Using the "condition - outcome" statement to express knowledge rules is very natural for humans. At the same time, in the Rule-based Expert System, knowledge and reasoning are stored and processed separately, in line with people's daily habits. However, some shortcomings also hindered the further development of such expert system. For example, when a rule matches a condition, the expression of the statement must be written strictly in accordance with the statement in the database, even if the relative subtle differences are rejected because of the matching accuracy requirements. In addition, the speed of the Rule-based Expert System is not dominant compared to other types of expert systems, since the entire rule set in the database needs to be scanned when a rule is used.

The structure of a rule-based expert system is shown in diagram below. The developer interface usually includes knowledge base editors, debugging aids and input/output facilities. All expert system shells provide a simple text editor to input and modify rules, and to check their correct format and spelling.

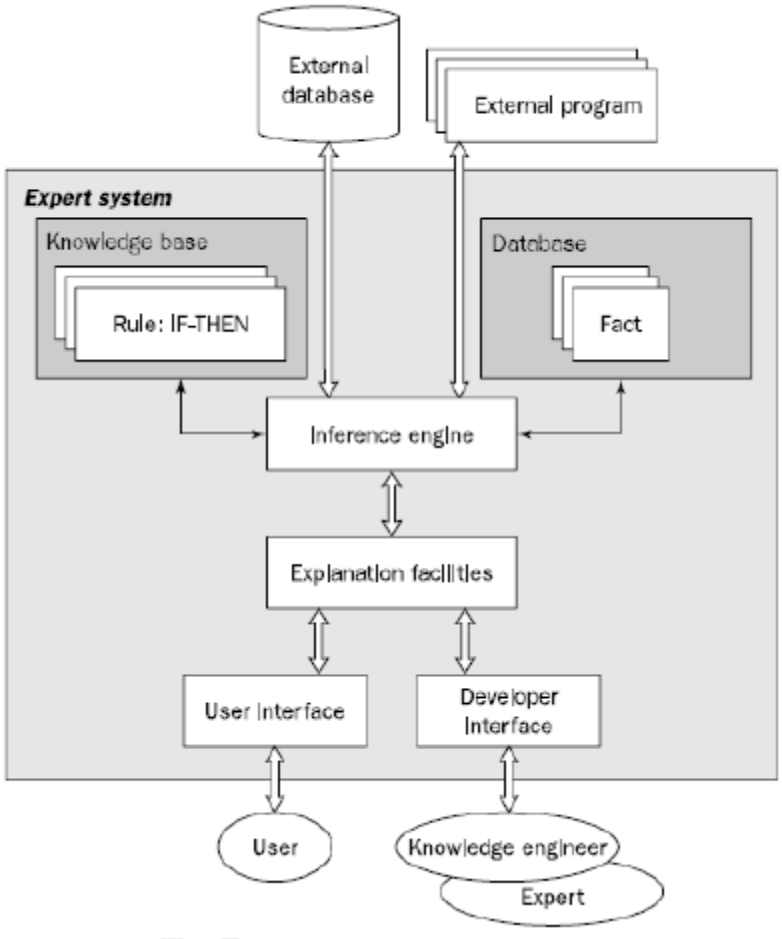

Figure 7: showing the complete structure of a rule-based expert system

3.1 Mycin- As an example of Rule Based Expert System

It is the name of a decision support system developed by Stanford University in the early- to mid-seventies, built to assist physicians in the diagnosis of infectious diseases. The system (also known as an "expert system") would ask a series of questions designed to emulate the thinking of an expert in the field of infectious disease, and from the responses to these questions give a list of possible diagnoses, with probability, as well as recommend treatment (hence the "decision support"). The name "MYCIN" actually comes from antibiotics, many of which have the suffix "-mycin".

The framework for MYCIN was derived from an earlier expert system called DENDRAL, created to find new chemical comounds in the field of orgainic chemistry (also developed at Stanford).

Components of Mycin

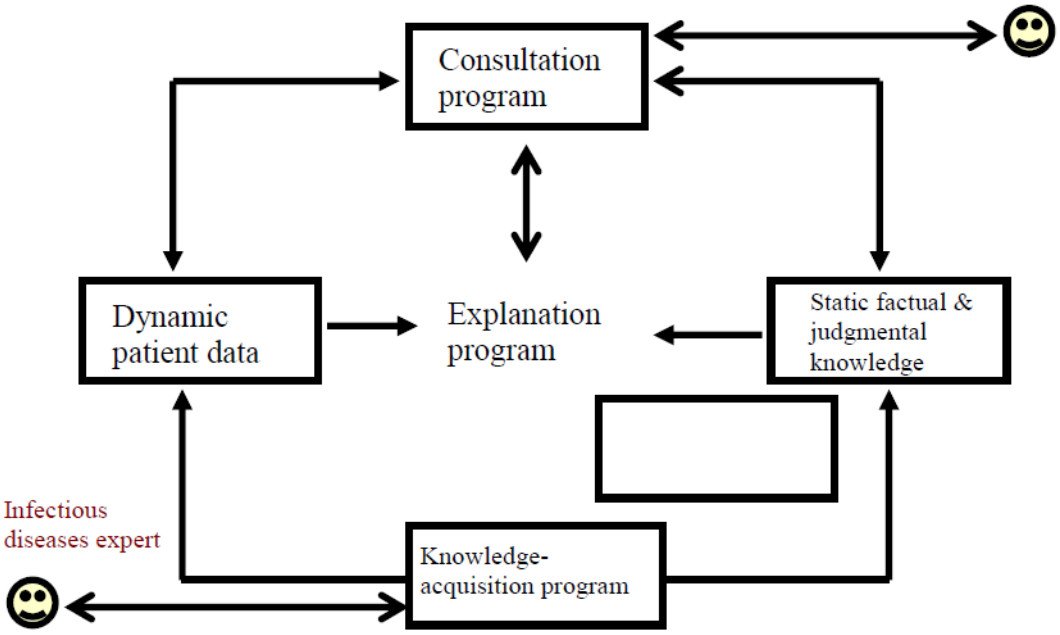

Figure 8: Component of Mycin system

The Knowledge Base

- Inferential knowledge stored in decision rules
  - If *Premise* then *Action* (*Certainty Factor* [*CF*])
  - If *A&B* then *C* (0.6)
  - The CF represents the inferential certainty
- Static knowledge:
  - Natural language dictionary
  - Lists (e.g., Sterile Sites)
  - Tables (e.g., gram stain, morphology, aerobicity)
- Dynamic knowledge stored in the context tree

- Patient specific
- Hierarchical structures: Patient, cultures, organisms
- <*Object*, *Attribute*, *Value*> triples: <Org1, Identity, Strep>
- A CF used for factual certainty <Org1, Identity, Staph, 0.6>

Mycin is an expert system comprised of two major components:

1. A knowledge base which stores the information the expert system "knows", much of which is derived from other information in the knowledge base.
2. An inference engine to derive knowledge from the presently known knowledge in the knowledge base.

Knowledge represented by MYCIN's knowledge base is represented as a set of IF-THEN rules with particualr certainty factors. For example:

IF the infection is primary-bacteremia
AND the site of the culture is one of the sterile sites
AND the suspected portal of entry is the gastrointestinal tract
THEN there is suggestive evidence (0.7) that the infection is bacteriod.

Humans interface with MYCIN by answering a series of diagnostic questions akin to what a physician may ask a patient, as well as prompting for relevant test results. MYCIN takes this data as input and either arrives at a set of answers with respective probabilities, or branches to other questions in order to narro it's search. Researchers at Stanford found MYCIN to have an approximate correctness rate of 65%, which is better than the majority of physicians who are not specialists in diagnosing infections, and only slightly worse than physicians who were experts in that field (who had an average correctness of approximately 80%).

3.1.1 Mycin's knowledge base

It is small relative to those used by most rules-based systems today; it is on the order of ~500 rules. The science of the generation of these rules is known as "knowledge engineering". MYCIN uses a modification of the method of reasoning called "backward chaning" to search it's knowledge base. The modification comes in when the system is in the beginning stages of diagnosis, when the system asks a series of broad questions in order to weed out any unnecessary searches later on, such as checking for pregnancy if the patient is male. Only when the problem becomes more defined does MYCIN use full backwards chaining. If MYCIN checks a rule with a probability of 0.2 or less, it will abandon further searching on that particular rule.

3.1.2 The Inference Engine

The core of MYCIN, its inference engine, is called EMYCIN ("Essential MYCIN"). EMYCIN is the framework for MYCIN, a semi-separate system which could be used to create other rules- based expert systems to face problems similar to what MYCIN faces. In some cases this can be affected merely by changing the knowledge base.

- **Which features make rule-based expert systems particularly attractive for knowledge engineers?**

Among these features are:

1- **Natural knowledge representation**. An expert usually explains the problem solving procedure with such expressions as this: 'In such-and-such situation, I do so-and-so'. These expressions can be represented quite naturally as IF-THEN production rules.
2- **Uniform structure**. Production rules have the uniform IF-THEN structure. Each rule is an independent piece of knowledge. The very syntax of production rules enables them to be self-documented.
3- **Separation of knowledge from its processing**. The structure of a rule-based expert system provides an effective separation of the knowledge base from the inference engine. This makes it possible to develop different applications using the same expert system shell. It also allows a graceful and easy expansion of the expert system. To make the system smarter, a knowledge engineer simply adds some rules to the knowledge base without intervening in the control structure.
4- **Dealing with incomplete and uncertain knowledge**. Most rule-based expert systems are capable of representing and reasoning with incomplete and uncertain knowledge.

Rule-based expert systems are not problem-free?
There are three main shortcomings:
1- **Opaque relations between rules**. Although the individual production rules tend to be relatively simple and self-documented, their logical interactions within the large set of rules may be opaque. Rule-based systems make it difficult to observe how individual rules serve the overall strategy. This problem is related to the lack of hierarchical knowledge representation in the rule-based expert systems.
2- **Ineffective search strategy**. The inference engine applies an exhaustive search through all the production rules during each cycle. Expert systems with a large set of rules (over 100 rules) can be slow, and thus large rule-based systems can be unsuitable for real-time applications.
3- **Inability to learn**. In general, rule-based expert systems do not have an ability to learn from the experience. Unlike a human expert, who knows when to 'break the rules', an expert system cannot automatically modify its knowledge base, or adjust existing rules or add new ones. The knowledge engineer is still responsible for revising and maintaining the system.

4.0 Conclusion
In this unit, we have been able to:
- Explain rule-based system
- Identify a type of rule-based system, Mycin
- Explain the components of mycin
- Advantages and disadvantages of mycin

5.0 Summary
A rule-based expert system is an expert system which works as a production system in which rules encode expert knowledge. Most expert systems are rule-based. It is an expert system based on a set of rules that a human expert would follow in diagnosing a problem. An example of a rule-based expert system is Mycin. The framework for MYCIN was derived from an earlier expert system called DENDRAL, created to find new chemical comounds in the field of orgainic chemistry (also developed at Stanford). Mycin has two major components, these are knowledge base and an inference engine.

There are some qualities that make rule-based expert system a choice for some many developers and users, these are:

1. Natural knowledge representation. 2- Uniform structure. 3- Separation of knowledge from its processing. 4- Dealing with incomplete and uncertain knowledge.

Rule-based expert systems have advantages and disadvantages.

6. Tutor marked assignments

- Explain the term rule-based expert system
- Explain the term Mycin and its components
- What are the features that makes rule-based system popular
- What are the advantages and disadvantages of rule-based expert systems

7. Further readings and other resources

Djailani, I. (2018). Rule based expert system. Link: https://www.youtube.com/watch?v=cmu20qsRcSY

Unit 2: The Framework-Based Expert System

1.0 Introduction

In the previous unit we explain rule based expert system as one class of expert system; in this unit we would explain frame based expert system.

2. Objectives

In this unit you should be able to:

- Understand the concept of frame based expert systems
- Terms associated with frame based expert system
- How demon is triggered

3.**0 Frame-based expert system**: A frame is a data structure with typical knowledge about a particular object or concept. Frames were first proposed by **Marvin Minsky** in the 1970s. Each frame has its own name and a set of **attributes** associated with it. *Name*, *weight*, *height* and *age* are slots in the frame *Person*. *Model*, *processor*, *memory* and *price* are slots in the frame *Computer*. Each attribute or slot has a value attached to it. Frames provide a natural way for the structured and concise representation of knowledge. A frame provides a means of organizing knowledge in **slots** to describe various attributes and characteristics of the object. Frames are an application of **object-oriented programming** for expert systems. The concept of a frame is defined by a collection of **slots**. Each slot describes a particular attribute or operation of the frame. Slots are used to store values. A slot may contain a default value or a pointer to another frame, a set of rules or procedure by which the slot value is obtained. Frames includes information about how to use the frame, information about expectations (which may turn out to be wrong), and information about what to do if expectations are not confirmed, and so on. Collections of such frames are to be organized in frame systems in which the frames are interconnected. Frame-based systems are knowledge representation systems that use frames, as their primary means to represent domain knowledge. Knowledge is organized in the form of chunks called frames. These frames are supposed to capture the essence of concepts or stereotypical situations. For example, being in a living room or going out for dinner, by clustering all relevant information for these situations together.

Frame Based Expert System is associated with the objects of interest and reasoning consists of confirming expectations for slot values. Such systems often include rules too. In a frame- based system, the inference engine also searches for the goal, or in other terms for the specified attribute, until its value is obtained. In a rule-based expert system, the goal is defined for the rule base. In a frame-based system, rules play an auxiliary role. Frames represent here a major source of knowledge, and both methods and demons are used to add actions to the frames. Thus, we might expect that the goal in a frame-based system can be established either in a method or in a demon.

- A demon has an IF-THEN structure. It is executed whenever an attribute in the demon‘s IF statement changes its value. In this sense, demons and methods are very similar, and the two terms are often used as synonyms.
- However, methods are more appropriate if we need to write complex procedures. Demons, on the other hand, are usually limited to IF-THEN statements.

The framework-based expert system structures as "frame". The framework contains the name of the concept, the label slot of the main attribute/feature, and the possible values of each attribute, or the process of capturing procedural information about the concept. When a particular instance

of the concept is encountered, the relevant eigenvalue of the instance is entered into the framework, which is called the instance. The framework can be used for reasoning. The framework contains multi-aspect information of a concept that can be used even if the information itself is not observed. For example, whether or not we can see the roots, because the "tree" contains the "root" of the label, so we can think that the tree has root.
When looking for a framework to describe the current instance, it will often fail to match the situation. Specific examples will correspond to some frame fragments that used to match the candidate frames. The framework is usually used when most of the structure of a frame can be matched, since each frame contains information that can allow mismatches of attributes (features) to increase the fault tolerance of the frame. In addition, we can also save the framework of the search process to optimize the direction of the test until we find the most appropriate framework.
The Framework-Based Expert System uses the framework in the database to handle the specific issues of the input and output new information through the inference engine. A framework represents a concept of "class", and a framework may be a "subclass" of another framework, such as the "man" framework is a subset of the "human" framework. Some subclass-and-class relationships are semantic types, such as "I am" + "New York", "New Jersey", "Los Angeles".
The subclass framework inherits all the properties of its parent class framework, eliminating the need to reenter the properties of the process. However, we should pay attention to some special circumstances: some subclasses and their father class may be in a common property differences. When a frame belongs to multiple frames at the same time, it also inherits the properties of all these frames.

3.2.1 Terms Associated with Frame Base Expert System

- **Frame:** A data structure with typical knowledge about a particular object.
- **Slot:** A component of a frame in a frame-based system that describes a particular attribute of the frame.
- **Attribute:** A datum associated with an entity.

3.2.2 Frames As A Knowledge Representation Technique

- **Frames**
- **Class-Frame:** A frame that represents a class.
- **Instance-Frame:** A frame that represents a instance.
- **Facet:** A component of a slot in a frame.

3.2.3 The Slots Of A Frame Can Include:

- Frame name
- Frame hierarchy: parent frame(s)
- An attribute value. This can be a number, a Boolean value, or symbolic (character).
- A default value for the attribute.
- The range of valid values for the attribute.

A subprogram which is executed when the slot value is changed or accessed. These subprograms are called demons in the AI world. (These are not the same as the demons, or daemons, of the UNIX world).

A demon can be triggered when:

- A new value is added to a slot.
- A value is removed from a slot.
- The value in a slot is replaced by a new value.
- A value is requested from an empty slot (which does not contain a value).

3.2.4 Demons are facets:

The term "frame" is used to refer both to class-frames, which are abstract templates, and to instance frames, which are concrete analogs of a particular entity.

3.2.5 Inheritance in Frame-Based Experts

Frames

In the context of frames, inheritance refers to:

- The inheritance by a class-frame of the characteristics (slots) of all its superclasses.
- The inheritance by an instance-frame of the characteristics of its class-frame.

4.0 Conclusion

In this unit you have learned:

- the concept of frame based expert system
- Terms associated with frame-based expert systems
- How a demon is triggered

5.0 Summary

In Frame-based expert systems, Frames provide a natural way for the structured and concise representation of knowledge. A frame provides a means of organizing knowledge in **slots** to describe various attributes and characteristics of the object. Frames are an application of **object-oriented programming** for expert systems. The concept of a frame is defined by a collection of **slots**. Each slot describes a particular attribute or operation of the frame. Slots are used to store values. A slot may contain a default value or a pointer to another frame, a set of rules or procedure by which the slot value is obtained. Frames includes information about how to use the frame, information about expectations (which may turn out to be wrong), and information about what to do if expectations are not confirmed, and so on. Collections of such frames are to be organized in frame systems in which the frames are interconnected.

A demon can be triggered when:

- A new value is added to a slot.
- A value is removed from a slot.
- The value in a slot is replaced by a new value.
- A value is requested from an empty slot (which does not contain a value).

6.0 Tutor-marked assignments

- Explain the term frame-based expert systems
- When is a demon triggered?

7.0 Further reading and other resources

Unit 3: Fuzzy logic and Neural-based expert systems

1.0 Introduction

Having explained rule-based and frame-based expert systems in unit 1 and 2, in this unit, we would step further to explain fuzzy and neural-based expert systems.

2.0 Objectives

In this unit, you should be able to understand:

- Fuzzy logic-based expert systems
- Neural network-based expert systems

3.0 *The Fuzzy Logic-Based Expert System*

In 1965, American mathematician Zadeh first proposed the concept of fuzzy set, marking the birth of fuzzy mathematics. Fuzziness refers to the indiscriminate nature of objective things or attributes, and there is a series of transitional states between them, without obvious dividing line. The fuzzy theory allows people to use mathematical tools to deal with the non-precise phenomenon in the real world.

In the Fuzzy Logic-Based Expert System, the fuzzy logic is the basis of expert system reasoning. This kind of reasoning method uses fuzzy rule as a prerequisite, and uses fuzzy language rule to deduce an approximate fuzzy judgment conclusion. The fuzzy language rules include the Generalized Modus Ponens (GMP) and the Generalized Modus Tollens (GMT).

The GMP rule can be expressed as: Prerequisite 1: x is A '. Precondition 2: If x is A, then y is B. Conclusion: y is B '. The GMT rule can be expressed as: premise 1: y is B, premise 2: If x is A, then y is B, Conclusion: x is A'. A, A', B, and B' in the above formula are fuzzy sets, and x and y are linguistic variables.

The fuzzy function A → B is considered as a "if - then" statement, forming a fuzzy reasoning rule.

The result of fuzzy reasoning is a fuzzy set, but in practical, a certain output value is required. The fuzzy decision is the process of obtaining a single value representing a fuzzy set. The simplest method is the maximum membership method, which takes the maximum value of membership in all fuzzy sets as output.

The advantage of the Fuzzy Logic-Based Expert System is that it can express the high skill of expert skills and has enough robustness, and can also carry out heuristic and tentative reasoning. However, this kind of expert system has difficulty in obtaining knowledge and its reasoning depends on fuzzy knowledge base, learning ability is not strong, prone to error.

3.1 The Expert System Based on Neural Network

The neural network or the artificial neural networks (ANN) are a biologically-inspired set of models that facilitate computers learning from observed data. The biological neurons are the basic motivation for the development of artificial neural networks. The brain has an estimated 100 billion neurons, each neuron has an average of 10 thousand connections that directly link it to other neurons. Thus there are about one million billion of these connections, making the brain “the most complex structure, natural or artificial, on earth” (Siegel, 2015). Electrical impulses transmitted over these connections create the interconnections in the brain; this web of interconnections means that activation of one neuron can influence an average of 10,000 neurons, creating an immense number of “on/off” patterns. An artificial neural network (ANN) is an information-processing system that adopts the neural structure of the human brain for analyzing data, finding patterns, classification, and prediction through a learning process using a series of mathematical equations. They are capable of decision-making byusing what they learn while encountering problems. Muller & Guido (2017)referred to neural networks as a family of algorithms which has recently seen a revival under the name “deep learning.” The deep learning concept shows great promise in many machine learning applications.

The neural network model is fundamentally different from the logical system described above. In the neural network, knowledge is no longer transformed into explicit rules by manual processing, but is automatically acquired by the learning algorithm and produces its own implicit rules. Compared with the traditional expert system, the neural network has more powerful function: it is more efficient than the traditional serial operation; it has a certain degree of fault tolerance; the weight of the neural network connection can be changed, etc.

The neural network acquires knowledge automatically through learning instances. Experts provide examples and expectations of the solution, neural network learning algorithm constantly modify the weight distribution of the network, achieving a stable output after training. Since the input and output of the neural network are numerical, it is necessary to encode the instance when using the neural network to acquire the knowledge.

The Expert System Based on Neural Network also has inherent weaknesses: the system performance is limited by the training sample set. In the case of improper sample set selection or too little sample, the neural network's induction reasoning ability is very poor. In addition, the neural network is unable to explain its own reasoning process and the significance of storing knowledge, because its model is based on human superficial neural activity. Different models can be applied to specific requirements of the expert system.

3.2 Can expert systems make mistakes?

Even a brilliant expert is only a human and thus can make mistakes. This suggests that an expert system built to perform at a human expert level also should be allowed to make mistakes. In building intelligent systems that mimic human expertise their performance metrics are not always 100%. But like human experts, we still trust their judgement, although we do recognize that their judgments are sometimes wrong. Likewise, at least in most cases, we can rely on solutions provided by expert systems, but mistakes are possible and we should be aware of this.

3.3 Does it mean that conventional programs have an advantage over expert systems?

In theory, conventional programs always provide the same 'correct' solutions. However, we must remember that conventional programs can tackle problems if, and only if, the data is complete and exact. When the data is incomplete or includes some errors, a conventional program will provide either no solution at all or an incorrect one. In contrast, expert systems recognize that the available information may be incomplete or fuzzy, but they can work in such situations and still arrive at some reasonable conclusion.

Another important feature that distinguishes expert systems from conventional programs is that knowledge is separated from its processing (the knowledge base and the inference engine are split up). A conventional program is a mixture of knowledge and the control structure to process this knowledge.

This inseparable logic and data sometimes leads to difficulties in understanding and reviewing the program code, as any change to the code affects both the knowledge and its processing. In expert systems, knowledge is clearly separated from the processing mechanism. This makes expert systems much easier to build and maintain. When an expert system shell is used, a knowledge engineer or an expert simply enters rules in the knowledge base. Each new rule adds some new knowledge and makes the expert system smarter. The system can then be easily modified by changing or subtracting rules.

The characteristics of expert systems discussed above make them different from conventional systems and human experts. A comparison is shown below.

Activity E: What are the classes of expert systems?

4.0 Conclusion

This Unit has discussed fuzzy and neural based expert system. We also clear some assumptions on expert systems as perfect system.

5.0 Summary

In this unit, we have learnt that:

Fuzzy and neural based expert systems.
Fuzziness refers to the indiscriminate nature of objective things or attributes, and there is a series of transitional states between them, without obvious dividing line. The fuzzy theory allows people to use mathematical tools to deal with the non-precise phenomenon in the real world. In the Fuzzy Logic-Based Expert System, the fuzzy logic is the basis of expert system reasoning. This kind of reasoning method uses fuzzy rule as a prerequisite, and uses fuzzy language rule to deduce an approximate fuzzy judgment conclusion.
An artificial neural network (ANN) is an information-processing system that adopts the neural structure of the human brain for analyzing data, finding patterns, classification, and prediction through a learning process using a series of mathematical equations.

6.0Tutor Marked Assignment

- Explain the term fuzzy logic based expert system
- Explain the term neural network based expert system
- Can expert system make mistake?

7.0 Further Reading and Other Resources

- ✓ Paliwal, M., & Kumar, U. . (2009). Neural networks and statistical techniques: A review of applications. *Elsevier: Expert Systems with Applications*, *36*, 2–9.
- ✓ Staub, S., Karaman, E., Kaya, S., Karapnar, H., & Güven, E. (2015). Artificial Neural Network and Agility. *Procedia - Social and Behavioral Sciences*, *195*(2015), 1477 – 1485.

- ✓ Lecture note by Younis, M.T on Rule-Based Expert Systems

Unit 4: Blackboard Expert System – HEARSAY

1.0 Introduction

The last unit exposed you to the different classes of expert systems. It also discussed the practical use of Mycin. In this unit, you will specifically learn about blackboard expert system with a particular emphasis on Hearsay

2.0 Objectives

By the end of this unit, you should be able to:

- Define a blackboard system
- Discus the components of blackboard system
- Identify the components of hearsay expert system
- Discus the benefit of blackboard architecture

3.0 Overview of Blackboard

A blackboard system is a task independent architecture for integrating multiple problem solving modules (referred to as knowledge sources). By task independent we mean that these architectures can be used for a wide range of tasks such as classification, design, diagnosis, repair etc. Integrated problem solvers, such as systems based on blackboard systems, attempt to overcome the inherent limitations of a single heuristic expert system by employing two or more problem solving (or reasoning systems). The problems solvers may all use the same technology or they may reply on different technologies. For example, a system might integrate a heuristic rule based reasoning system with a case-based reasoning system and possibly a model based system. However, integrated problem solving systems introduce a host of new issues. The overall problem facing such systems is that of integration of the problem solvers. Punch [1] categorizes integration techniques as being fixed, programmable or task specific. However, this is possibly too simplistic a view of the integration problem. Integrated systems can possess different features in different areas, for example, one system might possess a flexible programmable structure, yet have a fixed cooperation strategy. Integration really refers to the problems of control, cooperation and communication of diverse problem solvers within a single system Blackboard system.

3.1 Methods

In a blackboard system, a set of problem solving modules (typically called knowledge sources) share a common global database (called the blackboard). The contents of the blackboard denote and indeed are often called hypotheses. These hypotheses are often structured hierarchically. Knowledge sources respond to changes on the blackboard, and interrogate and subsequently directly modify the blackboard. This modification results form the creation, modification and solution of hypotheses. In fact only knowledge sources are allowed to make changes to the

blackboard. It is therefore through the blackboard that the knowledge sources communicate and cooperate. In blackboard architecture, each knowledge source responds only to a certain class or classes of hypotheses. These hypotheses, that a knowledge source responds to, often reflect the different levels in the blackboard‘s hierarchy. The blackboard holds the state of the problem solution, while the knowledge sources make modifications to the blackboard when appropriate.

3.2 Component of Blackboard

A blackboard system consists of three components:

• Knowledge sources (KSs) are independent modules that contain the knowledge needed to solve the problem. KSs can be widely diverse in representation and inference techniques.
• The blackboard is a global database containing input data, partial solutions, and other data that are in various problem-solving states.
• A control component makes runtime decisions about the course of problem solving and the expenditure of problem-solving resources. The control component is separate from the individual KSs. In some blackboard systems, the control component itself is implemented using a blackboard approach (involving control KSs and blackboard areas devoted to control).

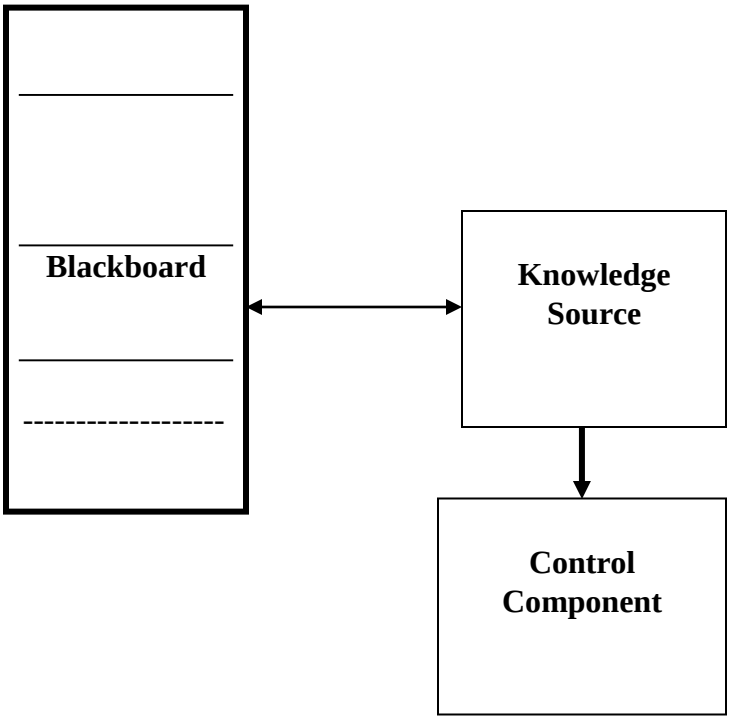

Fig 9: Basic Component of the Blackboard Model

3.3 Knowledge Source (KS)

Each KS is separate and independent of all other KSs. A KS needs no knowledge of the expertise, or even the existence, of the others; however, it must be able to understand the state of the problem-solving process and the representation of relevant information on the blackboard.

Each KS knows the conditions under which it can contribute to the solution and, at appropriate times, attempts to contribute information toward solving the problem. This knowledge that each KS has about when to contribute to the problem-solving process is known as a *triggering condition*.

KSs are much larger grained than the individual rules used by expert systems. While expert systems work by firing a rule in response to stimuli, a blackboard system works by firing an entire knowledge module, or KS, such as an expert system; a neural net or fuzzy logic routine; or a procedure.

Unlike our metaphor, KSs are not the active agents in a blackboard system. Instead, KS activations (sometimes called KS instances) are the active entities competing for execution resources. A KS activation is the combination of the KS knowledge and a specific triggering context. The distinction between KSs and KS activations is important in applications where numerous events trigger the same KS. In such cases, control decisions involve choosing among

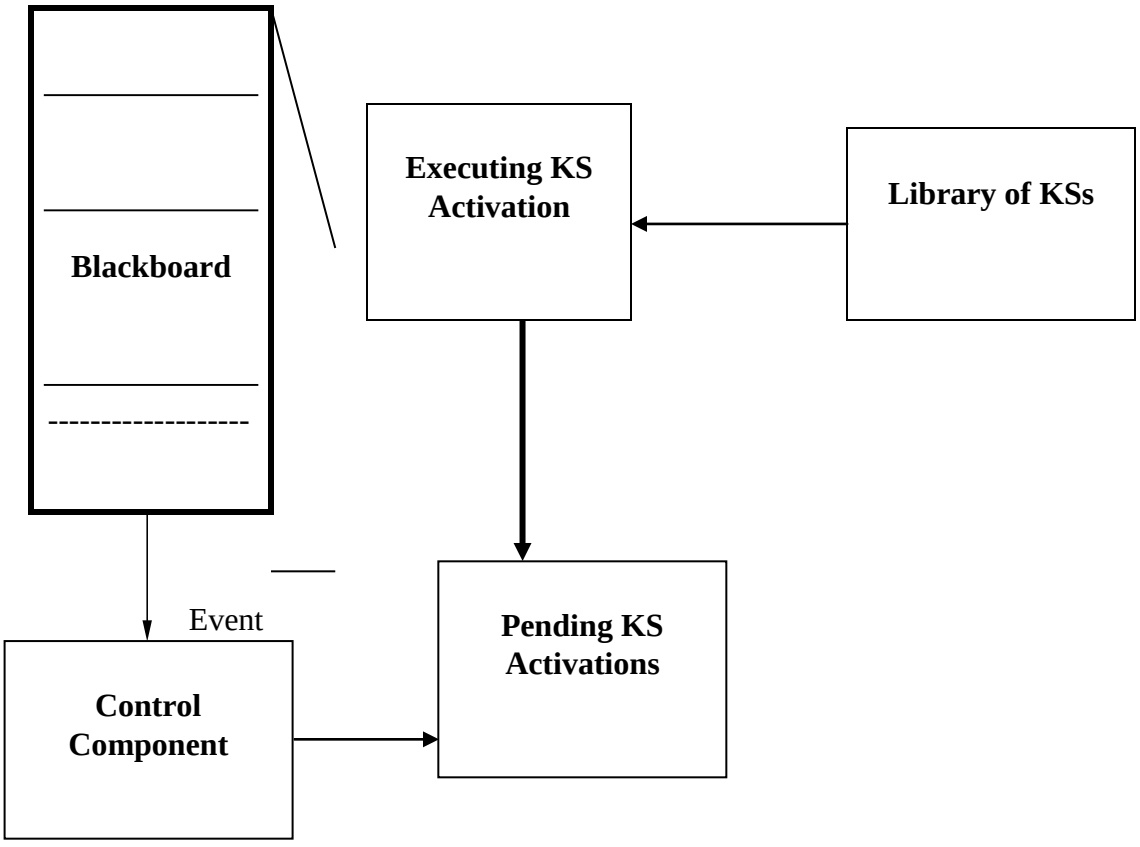

Fig 10: Basic Blackboard System

Particular applications of the same KS knowledge (focusing on the appropriate data context), rather than among different KSs (focusing on the appropriate knowledge to apply). KSs are static repositories of knowledge; KS activations are the active processes.

3.4 The blackboard

The blackboard is a global structure that is available to all KSs and serves as:

- a community memory of raw input data; partial solutions, alternatives, and final solutions; and control information
- a communication medium and buffer
- a KS trigger mechanism.

Blackboard applications tend to have elaborate blackboard structures, with multiple levels of analysis or abstraction.
Occasionally, a system containing subsystems that communicate using a global database is incorrectly presented as a blackboard system. (A set of FORTRAN routines using COMMON is an extreme example of this view). True blackboard systems involve closely interacting KSs and a separate control mechanism.

3.5 Control component

An explicit control mechanism directs the problem-solving process by allowing KSs to respond opportunistically to changes on the blackboard database. On the basis of the state of the blackboard and the set of triggered KSs, the control mechanism chooses a course of action.
A blackboard system uses an incremental reasoning style: the solution to the problem is built one step at a time. At each step, the system can:

- execute any triggered KS
- • choose a different focus of attention, on the basis of the state of the solution.

Under a typical control approach, the currently executing KS activation generates events as it makes contributions to the blackboard. These events are maintained (and possibly ranked) until the executing KS activation is completed. At that point, the control components use the events to trigger and activate KSs. The KS activations are ranked, and the most appropriate KS activation is selected for execution. This cycle continues until the problem is solved.
Blackboard systems support a variety of control mechanisms and algorithms, so a choice of opportunistic control techniques is available to the application developer.

The blackboard approach has been applied in numerous areas, including the following:

- sensory interpretation
- design and layout
- process control
- planning and scheduling
- computer vision
- case-based reasoning

- knowledge-based simulation
- knowledge-based instruction
- command and control
- symbolic learning
- data fusion

In each of these applications, the scope of the problem to be solved was the prime factor in selecting a blackboard approach. That is, deciding whether to use a blackboard approach should be based on the problem-solving requirements discussed above, rather than the specific application area.

3.6 The Hearsay Expert System

Hearsay,1985, separates the different types of knowledge into coherent modules, called knowledge sources (KSs) in his speech signal processing system. The independent knowledge sources for phonetics, syntax, semantics were independent processes that looked at the speech signal and posted hypotheses about likely syllables, words, phrases, and so forth on the blackboard – a global data-structure accessed by all of the KSs through an established protocol. Hypotheses generated by the lower level knowledge sources (about syllables and words) would be examined for feasibility by the syntactic and semantic KSs; these, in turn, could make suggestions about what words might be expected and post them on the blackboard. The advantages of this organization are those generally associated with modularization of knowledge. The blackboard organization for representing multiple types of knowledge has been used in several domains besides speech understanding.

3.6.1 Hearsay-II

In the Hearsay-II blackboard system, the flow of control occurs between the knowledge sources. These knowledge sources attempt to work in an opportunistic manner, influenced by the scheduler. By opportunistic we mean that cooperation occurs dynamically determined during the execution of the system based on the current information available.

In a blackboard system, each of the knowledge sources monitors the blackboard looking for an opportunity to aid in the emerging solution being recorded on it. If a knowledge source identifies a situation in which it can provide some information it registers its wish to be activated by posting a knowledge source activation record to the scheduler‘s agenda.

The scheduler determines which of the knowledge sources wishing to be activated next is chosen for activation. It does this using information in the focus-of-control database and the likely impact of the knowledge source. The impact might be the degree to which it refines the hypothesis space. The blackboard monitor is the system component that updates the focus-of-control database. It does this by monitoring the generation and modification of hypotheses on the blackboard. Such a control regime can be described as flexible control.

3.6.2 Architecture of Hearsay-II

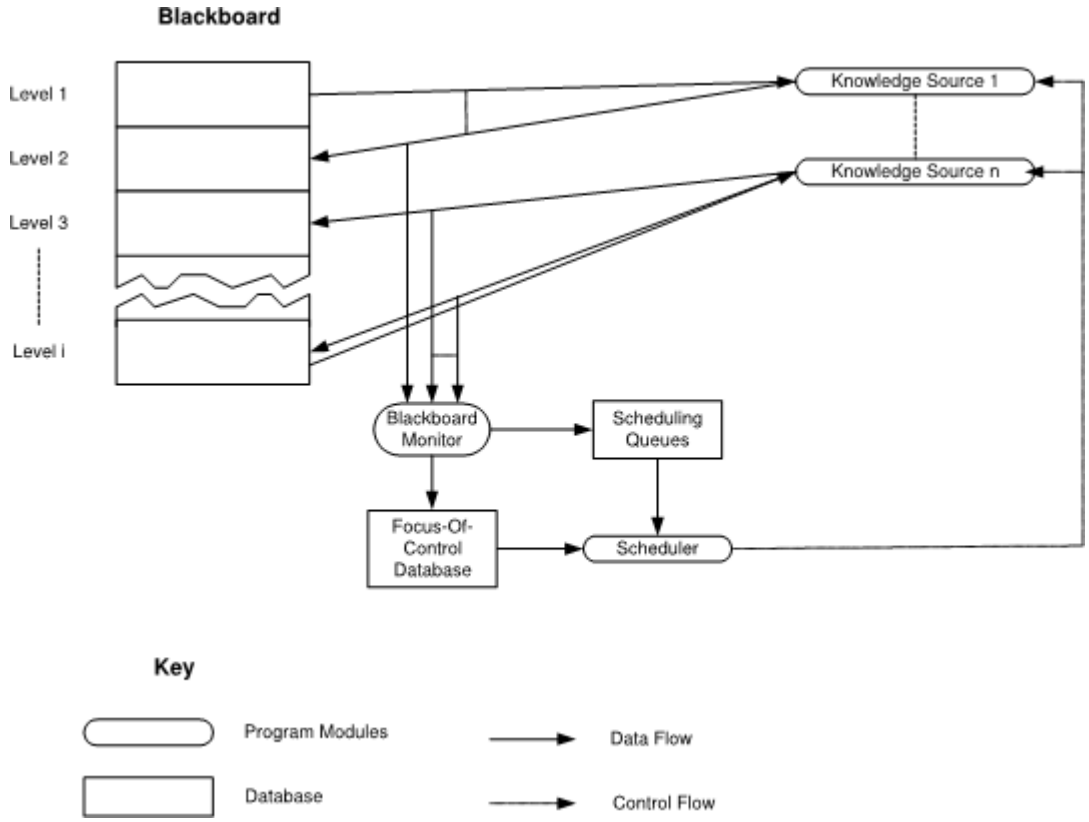

Fig 11: Architecture of Hearsay-II

3.7 Benefits of blackboard architectures

Some of the main benefits of blackboard architectures and the systems built using a blackboard framework are as follows:

- **Flexibility of configuration.** The knowledge sources within a blackboard system are not tied together in a fixed or rigidly applied manner. Instead they communicate and cooperate via the common blackboard. This means that any knowledge sources can be added to the system without having to specify its existence in any other knowledge source.
- **Flexible problem solving:** Blackboard architecture is independent of any particular task and can therefore be used as the basis of very different applications, from route planning, to image processing, speech recognition and diagnosis. Cooperation is driven by the current state of the problem, rather than through any particular control strategy, and control of this cooperation is distributed between the knowledge sources themselves.

These knowledge sources then determine which they are appropriate and the scheduler determines which of the appropriate knowledge sources to actually activate.

- **Selection of knowledge sources**. Because more than one knowledge source may be able to perform the same function, the scheduler can select that problem solver which will provide the most benefit to the emerging solution. This can improve both problem solving efficiency and the quality of the eventual solution.
- **Multiple problem solvers.** The blackboard architecture promotes the integration and multiple problem solving modules (knowledge sources). Each of these knowledge sources can potentially have its own representation and inference mechanism. This means that the twin advantages of multiple reasoning systems are maintained, namely: resilient behaviour and efficient behaviour.
- **Management of multiple levels of abstraction**. The blackboard architecture explicitly supports the management of multiple levels of abstraction. Not only can the blackboard be divided hierarchically, but the knowledge sources can reflect this hierarchy. Problems of moving between the levels of the hierarchy can be overcome by providing knowledge sources whose sole task is to move information between two particular abstraction levels.
- **Opportunistic cooperation**. Cooperation is a blackboard system is explicitly opportunistic; knowledge sources can post partial solutions to the blackboard in the hoped that some other knowledge source will be able to take these partial solutions and progress further down the path to the final solution.

3.8 Drawbacks of blackboard architectures

Some of the main drawbacks of blackboard architectures for integrating multiple problem solvers in a system such as a classification expert system include:

- **No communications language.** The blackboard acts as a very general communications facility. However, the communications language is the language of the hypotheses. That is, if all communications have to go via the blackboard then not only does the blackboard access mediate each decision step, it also constrains the representation of the information to be exchanged. That is, the language for passing information between problem solvers is the language of the intermediate hypotheses on the blackboard. This means that rather than an explicit request for some task to be performed, such as the generation of a simulation, the request must be phrased in the terms used to describe partial solutions on the blackboard – this may or may not be appropriate.
- **Computational complexity of cooperation.** The problem of determining which knowledge sources are appropriate in each situation and which of these appropriate knowledge sources should be activated has to be solved each time a decision has to be made. In the case of BB1 this can be computationally very expensive, even in Hearsay-II system this imposes a certain amount of computational overhead.
- **No guidance for task.** Blackboard architectures do not give any guidance to a system builder on how to solve a particular problem or perform a particular task within their framework. That is, blackboard architectures do not give any support for determining how specific problems should be addressed. Indeed they were never intended to do so, they are task independent architectures.
- **More complex problem solving framework.** The full facilities of a general problem

solving system are rarely required for a specific task. Yet blackboard architectures provide such a generality whatever the task being performed. Therefore, systems built within the blackboard framework often possess a more complex problem solving framework than is necessary.

- **No exploitation of task specific strategies.** There is no explicit support for task specific and domain specific problem solving strategies (for example, the use of useful diagnostic short cuts). Such strategies would have to be implicitly added into the scheduling system of the architecture.
- **Unbound nature of the control problem.** the potential unbound problem of determining which problem solver to invoke. A great deal of control problem solving‖ could occur before any real work is begun on solving the problem on the domain blackboard.
- **Global database.** The blackboard, through which all knowledge sources communicate, by which cooperation is achieved and on which the emerging solution is stored, is actually a global database. It is directly modified by the knowledge sources and assumes that knowledge sources will behave responsibly. If they do not, chaos could ensue. This assumption goes against all the principles of information hiding which have been developed by software engineers to promote well engineered systems.
- **Knowledge of problem solving scope**. Although individual knowledge sources are aware of the types of situations (problems) they can address, no knowledge of the scope of the system as a whole is maintained. Therefore blackboard systems are unable to easily detect problems on or beyond the periphery of the whole system‘s ability.
- **Knowledge acquisition.** Although the use of specialist knowledge sources should make it easier to develop problem solving systems, and to encode the appropriate knowledge in them, no support is given to help in the process of obtaining this knowledge in the first place. In addition, each knowledge source can potentially possess its own knowledge representation that may require its own particular techniques not only to obtain but also to codify.
- **Implicit representation of management information.** Within the scheduler of the Hearsay-II system, problem management information is implicitly represented. This information focuses on the generation of a solution, and determines which knowledge source, from those capable of processing a particular solution, should be invoked next. In order to be able to do this efficiently, knowledge about problem solving strategies, short cuts, the abilities of the knowledge sources and task specific techniques are required.

The implicit nature of the scheduler hides much of this information and some of these functions.

- **Problem solvers defined at system build time**. The knowledge sources need to be connected to the blackboard system at system build time. The abilities of the knowledge sources also need to be made available to the scheduler prior to the problem solving, in order that it can determine the –impact‖ of knowledge source.

Activity F What is a blackboard system?

4.0 Conclusion

In this unit, we have learnt about blackboard system, its architecture with a particular emphasis on hearsay expert system.

5.0 Summary

In this unit, we have learnt that:

- A blackboard system is a task independent architecture for integrating multiple problem solving modules (referred to as knowledge sources). By task independent we mean that these architectures can be used for a wide range of tasks such as classification, design, diagnosis, repair etc.
- A blackboard system consists of three components:
    1. Knowledge sources (KSs) are independent modules that contain the knowledge needed to solve the problem. KSs can be widely diverse in representation and inference techniques.
    2. The blackboard is a global database containing input data, partial solutions, and other data that are in various problem-solving states.
    3. A control component makes runtime decisions about the course of problem solving and the expenditure of problem-solving resources. The control component is separate from the individual KSs. In some blackboard systems, the control component itself is implemented using a blackboard approach (involving control KSs and blackboard areas devoted to control).
- Benefits of blackboard architectures include: Flexibility of configuration, Flexible problem solving, Selection of knowledge sources, Multiple problem solvers, Management of multiple levels of abstraction, Opportunistic cooperation.

- Drawbacks of blackboard architectures include**:** Problem solvers defined at system build time, Implicit representation of management information, Knowledge acquisition, Knowledge of problem solving scope, No exploitation of task specific strategies, More complex problem solving framework, No guidance for task, Computational complexity of cooperation, No communications language

6.0 Tutor Marked Assignment

- What is a blackboard system?
- Discus the architecture of a blackboard system
- Discus the components of blackboard system
- Discus the components of hearsay expert system
- Explain the benefit of blackboard architecture

7.0 Further Reading and Other Resources

Unit 5: Selecting Expert Systems-Based Tool (ESBT) for use in an Organization

3.0 Introduction

This unit reveals the understanding on how to select an expert system in an organization. It unveils the step by step process of selecting

2.0 Objectives

In this unit, you should be able to:

- To understand the how to select an expert system for an organization
- Know the criteria of selecting an expert system

3.0 Factors in selecting an expert system or expert –based tool

Perhaps the most important consideration in selecting an Expert based technology is the characteristics of the application (or applications) the system will be used to apply. Also important are the capabilities and skills of the developers or users who will use the tool or the system in the organization.
During the selection process, the characteristics of both application and developer are compared to capabilities of different tools in an effort to find an appropriate match.
Today, many commercial expert system-based tools are available for the microcomputer environment. These tools are complex, and have many features. It is therefore helpful to impose some structure on the process of matching characteristics of applications and developers to those of candidate tools. One way to structure the analysis is to divide it on the basis of the major areas of the ESBT architecture. The areas to consider in selecting ESBT are discussed below:

- Representing Knowledge About the Problem

It must be possible for the ESBT to represent knowledge about the problem and the way it is solved. In some cases, only production rules are needed. For complex applications, pattern matching, frame systems, or object-oriented programming may also be necessary.
.

- Selecting an Appropriate Inference Strategy determining whether backward chaining, forward chaining, or mixed inference is required is equally important.
- Determining Characteristics of End Users for expert system applications with human end users, the requirements (and preferences) of end users are important factors. The developer must determine what kinds of computer Interfaces are needed and select an ESBT that can provide them. For example, if end users are accustomed to seeing diagrams, it may be appropriate to select an ESBT that supports graphics capabilities.
- Determining Requirements for External Interfaces

If the application must interface with other software systems, the ESBT must provide appropriate interfaces.

- Selecting the Developer Interface

The development interface supported by the ESBT must be appropriate for the needs and skills of the developer. Important considerations include ease of use and ease of learning.

The actual analysis may vary in specific situations, depending on the characteristics and requirements of applications, developers, and the organizations they belong to. In addition, the analysis must take into consideration other factors. A particularly important consideration is the microcomputer platform the application will be developed on and the platform the application will be delivered on. (They may be the same or different.) The ESBT must be able to run on the platform intended for development and must also support delivery of applications on the intended platform.

The analysis must also consider the level of performance provided by the ESBT. Performance depends on both the efficiency with which the ESBT was implemented and the power of the microcomputer platform upon which the application will run.

3.1 Selection Steps

The actual process of selection might be described in a sequence of steps, summarized below.

- Determine the characteristics of the expert system application or applications.
- Identify the characteristics of the developers, particularly in regard to level of software development skill.
- Assess the hardware platform (or platforms) that will be used for application development and for delivery of the completed system. This analysis may be limited to hardware already available in the organization or involve purchase of new computers. Relevant issues include performance, ease of use, product reliability, and copyright issues. A particularly important consideration is the operating system (or systems) that run on the computer. The ESBT and the operating system must be compatible.
- Create a list of selection criteria based on knowledge of application characteristics, developer characteristics, and hardware characteristics.
- Specify the list to identify mandatory features and optional features that are desirable (as well as features that may sometimes be supported but are not needed or wanted)

.

- Research individual products. Information about products can be obtained by:

(1) Surveying available literature.
(2) Attending computer conferences in which ESBT vendors display products.
(3) Attending product seminars presented by vendors.
(4) Contacting current users of ESBTs.
(5) Direct examination and evaluation of commercial ESBTs.

Direct examination and evaluation of ESBTs require access to and use of individual products for a period of time. If a sample of the target application has been developed, it can be used for comparative benchmarking.

- The final selection is based on an overall comparative rating of features.

Once the ESBT is selected, further information about the tool will be acquired after development begins. Often, prototyping brings to light previously unknown characteristics of the application, sometimes making it necessary to change tools. In some cases, it may be desirable to prototype using one ESBT and use a different tool for large-scale development and to deliver the completed system.

3.2 Level of Effort in the Selection

The level of effort involved in the selection process will vary depending on the size of the application and the budget of the particular project. It would not be cost-effective to allocate hundreds of staff hours to select an ESBT that costs a little. Similarly, it would not be cost-effective to make a hasty decision on a software package costing thousands of dollars. If the requirements of the application can be satisfied by inexpensive tools, it may be cost-effective to purchase several tools and experiment with them.

Other Considerations in Selection besides the technical feature-by-feature evaluation, there are other considerations relevant to the selection process. These areas of concern are discussed briefly below:

- Portability

Though no standards have been developed specifically for ESBTs, portability across platforms is an effective means of maintaining long-term, stable usage of the selected ESBT.

- Vendor Support

Evaluation of vendor support should consider such factors as comprehensiveness of manuals, availability of training, "hotline" support, and warranties for the software product, etc.

- Reliability and Product Bugs

It is advisable to procure an ESBT that has been available for a period of time and has had many users. This helps ensure that as many bugs as possible will have been discovered and fixed. New versions of the ESBT should be downward compatible. This helps assure that the introduction of a new version of an ESBT product does not require extensive revision to operational applications.

- Licensing Considerations

Questions about the cost of site licensing (the cost of obtaining a copy of the ESBT for a particular site) and runtime licensing (the cost of each runtime copy of a developed expert system application) may also be important.

During the selection process, there are many trade-offs. The desirability of different features must ultimately be weighed against the cost of the tool that supports them. Frequently no single ESBT supports all desired features. For instance, a particular tool may support a sophisticated development environment, but it may lack the inference strategy appropriate for the intended application. In such cases, a sacrifice may be necessary, or an alternate means of implementing a capability must be sought.

Activity E: What are the steps to be taken in selection an expert system-based tool?

4.0 Conclusion

In this unit, we have been able to reveal the process of selecting an expert system based tool in an organization. We also look at the steps to be taken as well as the criteria to note.

5.0 Summary

- The factors in selecting ESBT includes: Representing Knowledge About the Problem; Selecting an Appropriate Inference Strategy determining whether backward chaining, forward chaining, or mixed inference is required is equally important; Determining Characteristics of End Users for expert system applications with human end users; Determining Requirements for External Interfaces; Selecting the Developer Interface.
- We also look at selection Steps, these are: Determine the characteristics of the expert system application or applications; Identify the characteristics of the developers; Assess the hardware platform (or platforms) that will be used for application development and for delivery of the completed system; Create a list of selection criteria based on knowledge of application characteristics, developer characteristics, and hardware characteristics; Specify the list to identify mandatory features and optional features that are desirable; Research individual products.

6.0 Tutor Marked Assignment

- What are the factors in selecting ESBT in an organization
- .What are the selection steps of ESBT in an organization

7.0 Further Reading and Other Resources

Unit 6: Expert System Shells

1.0 Introduction

In the last unit, you learnt about how to select an expert system based tool in an organization. In this unit, you will be exposed to Shells, an example of an expert system development tools and its components.

2.0 Objective

In this unit, you should be able to:

- Define Shells
- Explain the components of shell

3.0 Overview of Shells

An expert system shell can be considered as an expert system with the knowledge removed. Therefore, all the user has to do is to add the knowledge in the form of rules and provide relevant data to solve a problem.

A shell (or inference engine) is a complete development environment for building and maintaining knowledge-based applications. It provides a step-by-step methodology, and ideally a user-friendly interface such as a graphical interface, for a knowledge engineer that allows the domain experts themselves to be directly involved in structuring and encoding the knowledge. Examples of shells include Drools, CLIPS, JESS, d3web, G2, eGanges, and OpenKBM (initially developed as a replacement for G2).

Expert system shells provide methods of building expert systems without extensive knowledge of programming through mechanisms that (1) input the decisions, questions and rules that are followed (2) construct a knowledge database that can be manipulated by subsequent parts of the system (3) verifies possible violations of surface validity and (4) operates the "inference engine" that operates on the rules, poses the questions to the users, and determines whether a particular decision is valid. Most expert systems also allow the user to halt the processing at any time to query the system why a question was asked, or how a decision was reached.

Most expert system shells can now run easily on most current micro-computers and are able to handle the manipulation of a relatively large number of rules and associated questions.

Expert system shells are expert system development tools consisting essentially of the expert system without the knowledge base, embodying the inference engine, working memory, and the user interface.

Expert systems give advice or solve problems by drawing upon this knowledge stored in the IF/THEN rules.

An Expert system shell is a software development environment. It contains the basic components of expert systems. A shell is associated with a prescribed method for building applications by configuring and instantiating these components.

3.1 Shell components and description

The generic components of a shell are: the knowledge acquisition, the knowledge Base, the reasoning, the explanation and the user interface are shown below. The knowledge base and reasoning engine are the core components.

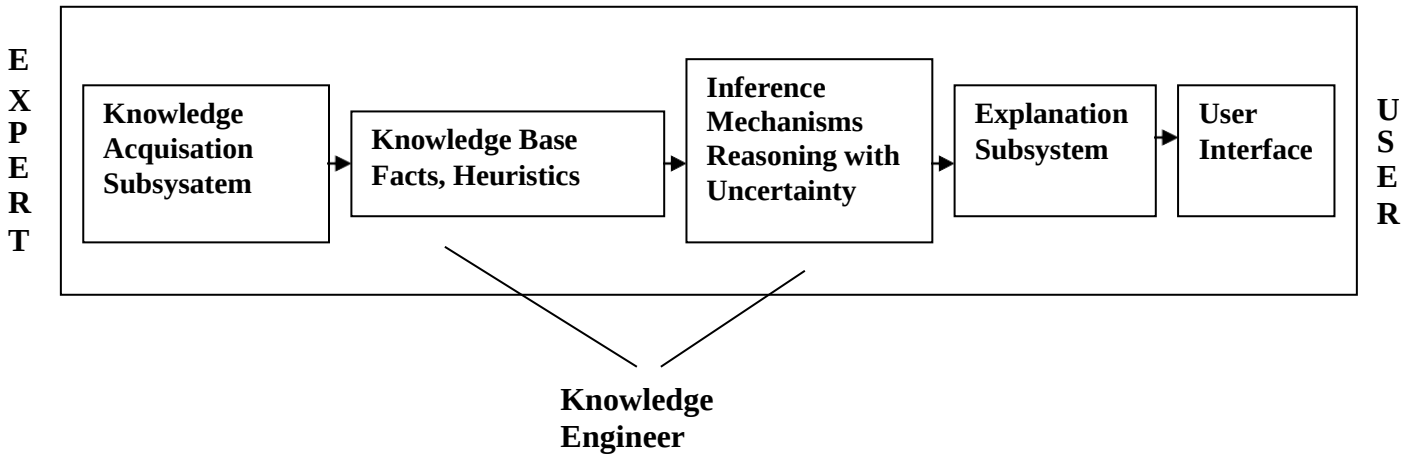

Fig 16: Shell Component

3.1.1 Knowledge Base

It is a store of factual and heuristic knowledge. Expert system tool provides one or more knowledge representation schemes for expressing knowledge about the application domain. Some tools use both Frames (objects) and IF-THEN rules. In PROLOG the knowledge is represented as logical statements.

3.1.2 Reasoning Engine

Inference mechanisms for manipulating the symbolic information and knowledge in the knowledge base form a line of reasoning in solving a problem. The inference mechanism can

3.1.3 Knowledge Acquisition subsystem

A knowledge subsystem helps experts in building knowledge bases. However, collecting knowledge, needed to solve problems and build the knowledge base, is the biggest bottleneck in building expert systems.

3.1.4 Explanation subsystem

An explanation subsystem explains the system's actions. The explanation can range from how the final or intermediate solutions were arrived at justifying the need for additional data.

3.1.5 User Interface

A user interface is a means of communication with the user. The user interface is generally not a part of the expert system technology. It was not given much attention in the past. However, the user interface can make a critical difference in the perceived utility of an Expert system.

3.1.6 Explanation

Most expert systems have explanation facilities that allow the user to ask questions - why and how it reached some conclusion. The questions are answered by referring to the system goals, the rules being used, and existing problem solving. The rules typically reflect empirical or "compiled" knowledge. They are codes of an expert's rules of thumb, not the expert's deeper understanding.

Activity H: What is a shell?

4.0 Conclusion

In this unit, you have learnt about shell and its various components

3.1 Summary

In this unit, you have learnt that:

3.1.1 A shell (or inference engine) is a complete development environment for building and maintaining knowledge-based applications.

3.1.2 The generic components of a shell are the knowledge acquisition, the knowledge Base, the reasoning; the explanation and the user interface. The knowledge base and reasoning engine are the core components.

6.0 Tutor Marked Assignment

- What is a shell?
- Discus the various components of shells system.

7.0 Further Reading and Other Resources

1. Rich, E and Knight, K. (2006). Artificial Intelligence. McGraw Hill companies Inc., Chapter 20, page 547-557.

2. Patterson, D.W. (2007). Introduction to Artificial Intelligence & Expert Systems, Prentice-hall.

.

3. Krishnamoorthy, C.S Artificial intelligence and expert systems for engineers.

Module 3: Current Trends in Expert Systems
Unit 1: New development in expert systems

1.0 Introduction

This discusses the current trend in expert system design and applications in different areas of human endeavor.

2.0 Objectives

By the end of this unit, you will be able to:

- Discus the current trend in expert systems
    - 2.1.1 Understand the industry with wide expert system application

3.0 The new development in expert systems

The limitations of the previous type of expert systems have urged researchers to develop new types of approaches. They have developed more efficient, flexible and powerful approaches in order to simulate the human decision-making process. Some of the approaches that researchers have developed are based on new methods of artificial intelligence (AI), and in particular in machine learning and data mining approaches with a feedback mechanism. Modern systems can incorporate new knowledge more easily and thus update themselves easily. Such systems can generalize from existing knowledge better and deal with vast amounts of complex data.

With the rising of Machine learning and statistical learning, there are now new research hotspot in artificial intelligence. Naturally, people thought that if the machine learning technology applied to the expert system, the reasoning performance of the expert system would be promoted. The fuzzy logic, and deep learning are new area that are being applied to the expert system development. Now, all kinds of different expert system blossom, expert system research has entered a new period of prosperity. Today, the demand for expert systems is further enhanced, and people are eager to use expert system to solve more complex problems.
It is important to reiterate that the earlier predominant programming tools such as Lisp and Prolog are not in regular usage any more.

3.1 Expert Systems by Functional Area

From the organizational viewpoint, most of the expert applications were from business domains.
A survey research conducted in over 33 years of expert systems applications shows that there was a major spike of interest in expert systems in the late 80s and early 90s mainly in Operations (including Scheduling applications), Finance, and Management. There was also some interest in applications for Production (manufacturing) and Accounting. After this spike in interest however, applications were generally limited to operations and general management. This may be due to the fact that the majority of interest in expert systems has come from areas with well-structured problems such as operations or finance that can be more easily programmed as expert system applications. The chart below shows the representations of growth and interest in some functional areas of expert systems applications.

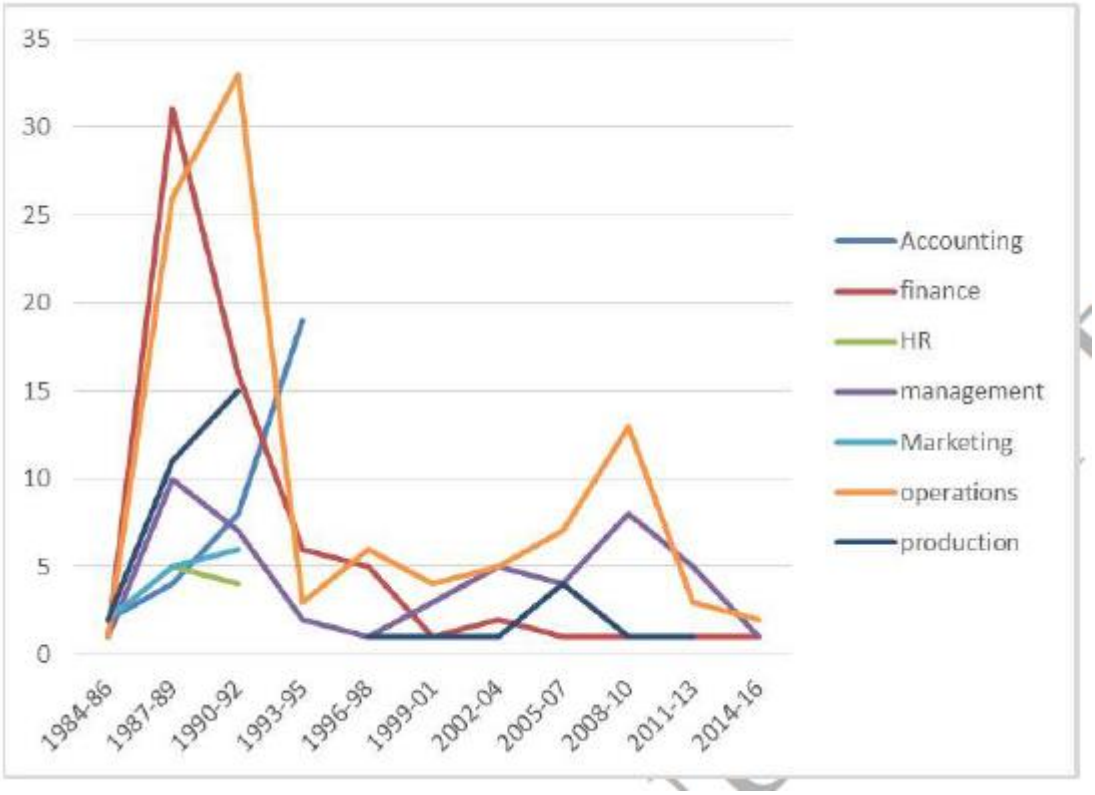

Figure: Expert System Applications by Functional Area over Time

3.2 Expert Systems by Industry

A survey of industries conducted between 1984 and 2016 to understand specific industry were expert systems have been applied showed that the popular industries for expert system applications were accounting, financial services, manufacturing and medicine. These are industries that are quick to explore and adopt new technologies such as expert systems. They also typically devote more resources to information technology since they have a greater potential payback. The industry laggards were publishing, real estate, and legal services. It was a little surprising that the transportation industry was also on the low end of the industry applications.

3.3 Problem Domain Trends

The concept of a problem domain describes the general focus or issue of the particular application. Expert systems have to have a very well-defined problem domain in order to function; this is an especially important concept and one of the major determinants of expert system quality. In problem domain delineation, analysis of the problems involve breaking it down into sub-problems. Expert systems are readily applied in easily structured problems.
The chart below shows the problem areas were experts system are applied

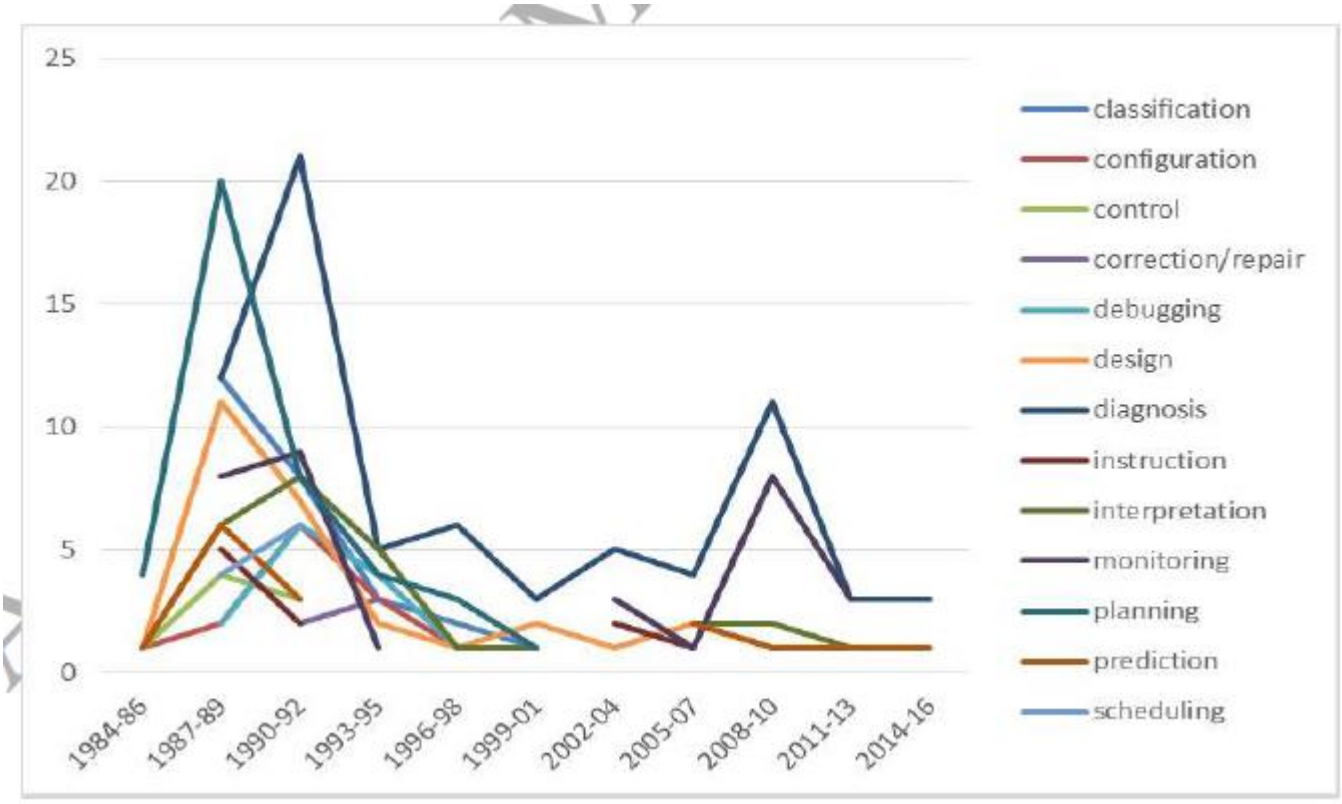

Figure: Expert Systems Problem Domains over Time

3.4 Trends in Knowledge Acquisition Techniques

At the outset of expert systems, it was often the case that the domain expert and the knowledge engineer were one and the same. So at that time there was little discussion about the knowledge acquisition (KA) and it attendant bottleneck.

In the past, researchers either borrowed interviewing techniques from the field of psychology, or wrote programs that helped automate and structure the interview process. The manual process was dominantly used in knowledge acquisition, the process involve the structured interview in the knowledge engineering. Tools such as surveys, focused questions, knowledge maps and even the prototype systems themselves to help structure the interview. Since they were popular at the time, some other specific KA techniques such as protocol analysis (verbal, video, and textual) along with KE observation.

After a while, there was much more of a focus on using the appropriate technique for knowledge acquisition Empirical research in the field of expert systems revealed that certain KA techniques are significantly more efficient than others in different problem domains and KA scenarios. While working on projects some of the categories of KA were updated to include the possibility of computer learning techniques such as ANNs, Bayes Networks and also case-based reasoning being integrated into the project. Computer modelling was expanded to include a variety of computer generated models such as repertory grids, multi-dimensional scaling and semantic networks.

The chart below shows the different knowledge acquisition used in expert systems.

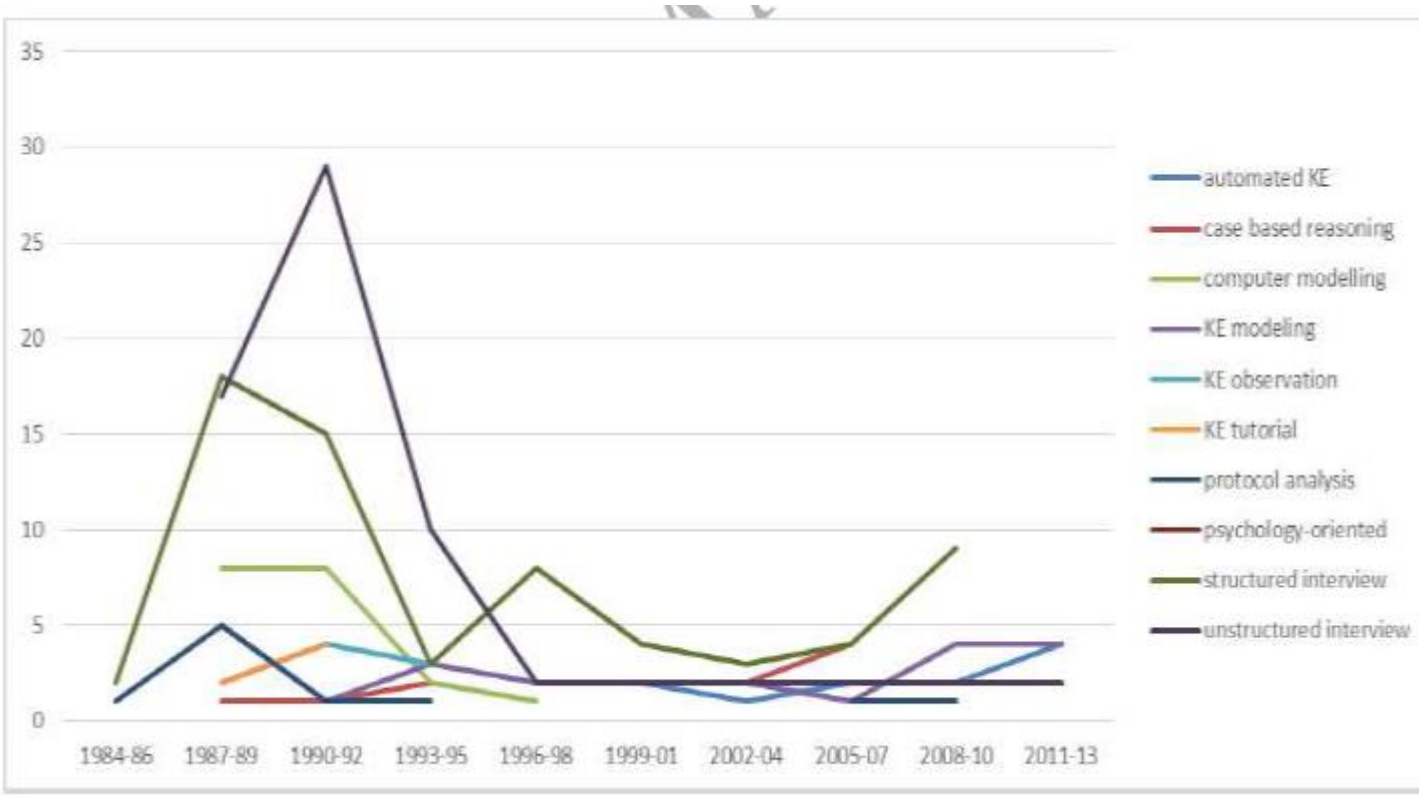

Figure: Expert System KA Techniques over Time

3.5 Knowledge Representation Trends

One of the more difficult determinants to accurately track is that of the particular knowledge representation (KR) schema used in expert system development. While certain knowledge acquisition techniques and expert system platforms may lend themselves to a particular KR schema, in practice there may be many different intermediate KRs used. These could include semantic nets, ontologies, decision trees, cases, neural nets, etc. A target or primary KR schema was identified and additional intermediate KRs were tracked separately when they were included in the case. The different types of KR schema used in expert systems over time are shown in the chart below.

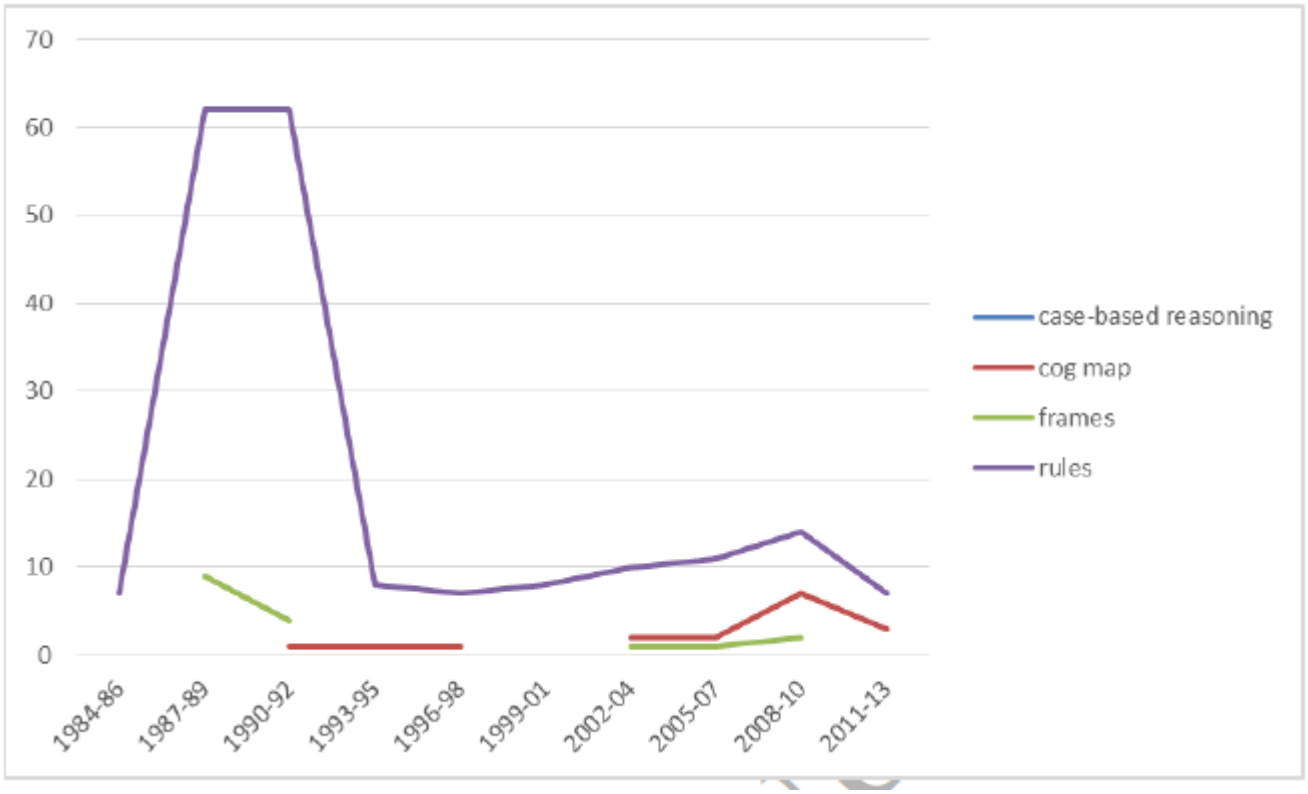

Figure: Expert Systems by KR Schema over Time

4.0 Deploying Expert Systems within the Organization

Determining how and where to use expert systems are based on analyzing the information processing needs of an organization. Identification of places to apply expert system technology is based on finding critical points within an organization where automation of expertise can lead to improvements in operational efficiency. These are some of the reasons that necessitates the adoption and deployment of expert systems in an organization.

- Alleviating "Knowledge Bottlenecks"

"Knowledge bottlenecks" occur when existing expertise cannot be brought to bear on regularly occurring problems that require expertise to solve. That is, "knowledge bottlenecks" happen when the number of experts is too small for the number of problems that need to be solved. Or experts may be geographically distant from the site of the problem.

- Providing a Means for Consistent Decision Making

By deploying expert systems throughout an organization, expertise about narrowly focused problems can be disseminated. This can be used to ensure consistent implementation of policies and procedures.

- Automating Repetitive Tasks That Are Difficult for Humans

In many operational situations, certain tasks are performed by continuous repetition over long periods. One example is monitoring computer systems for illegal access attempts. Such tasks require a certain level of expertise, but they are performed poorly by people because they require constant attention. In these situations, expert systems have proved useful.

- Freeing Experts for More Important Tasks

An expert system can perform many relatively mundane tasks normally handled by an expert. This allows the expert to devote time to other work that may also be important to the organization.

Activity I: Explain the new trends in expert systems?

5.0 Conclusion

In this unit, you have learnt about current trends in expert systems

6.0 Summary

In this unit, you have learnt that:

- The new development in expert systems
- The functional area of expert systems applications

7.0 Tutor Marked Assignment

- Why are there new developments in expert systems?
- Explain the new developments in expert system

8.0 Further Reading and Other Resources

- Arcucci, R. (2020). Machine learning and data assimilation for expert systems. Link: https://www.youtube.com/watch?v=l-603fZUlB8