Journal of Physics: Conference Series

PAPER • OPEN ACCESS

Design of an expert system architecture: An overview

To cite this article: Durgaprasad Janjanam *et al* 2021 *J. Phys.: Conf. Ser.* **1767** 012036

ECS
240th ECS Meeting
Digital Meeting, Oct 10-14, 2021
We are going fully digital!
Attendees register for free!
REGISTER NOW

Design of an expert system architecture: An overview

Durgaprasad Janjanam[1], Bharathi Ganesh[2] and Manjunatha L[3]

[1,2] Professor, Nitte Meenakshi Institute of Technology, Bangalore.
[3]Asst. Professor, Nitte Meenakshi Institute of Technology, Bangalore.

Email: durgaprasad.j@nmit.ac.in

Abstract. An area of Artificial intelligence (AI) that can claim a large measure of responsibility for the current heightened AI awareness is Knowledge Based Expert System (KBES), computer programs that embody human expertise. KBES specifically enable the computer to assist in solving ill-structured and non-deterministic problems such as design activities including synthesis, evaluation, modeling and decision-making process. The paper explains the role of Backward Chaining method, Forward Chaining method, Rule-Value method as three major methods involved in solving these problems to differentiate the Knowledge Based Expert System programming from normal programming. The development of an expert system requires a tool that will provide for better means of knowledge acquisition, inference mechanism and user interface. This paper brings out an overview of Expert System, a systematic study involving a critical analysis of the Expert System (ES) development, specifying its requirements. Further it highlights the features of Expert System and its implementation.

1. Introduction

Artificial intelligence (AI) concerned with designing intelligent computer system that changes something from useful to something essential [1], [2]. One area of AI that can claim a large measure of responsibility for the current heightened AI awareness is Knowledge Based Expert System (KBES), computer programs that embody human expertise. It is one of the important research fields in Artificial Intelligence for problem solving. It is the first AI technology to widely impact business and industry. It offers much practical use and commercial potential [3]. There are many applications that have comprehensive problem-solving tasks of an algorithmic or deterministic nature, such as evaluating, testing, scheduling, and calculating and presenting graphical. Nevertheless, the theory behind decision-making processes and design practices such as synthesis, assessment, and simulation of decision-making processes are ill-structured and involve judgment and experience-based heuristic solution [4]. KBES helps the computer to assist in the mismanaged and non-deterministic problem-solving activities such as a significant dimension to the cycle of design and decision taking. An efficient shell is a pre-requisite in developing the KBES. The shell will have facilities for easy information implementation, and for smooth interaction between the user and the network of experts. The core units of shells should include module for information acquisition, inference mechanism with correct user interface.

This paper contains an overview of Expert Systems, a system study involving a critical analysis of the Expert System (ES) development, specifying its requirements. Further, it highlights the features of the future ES and its implementation.

2. Expert System – An overview

2.1 Introduction

An Expert System is a computer program designed to act as an expert in a particular area or domain of expertise. Often referred to as an information-based expert system, an expert system typically includes a robust knowledge base composed of domain facts and heuristics, or guidelines for applying these facts [5].

a) KBES has proven useful in solving a variety of field problems, including medical diagnosis, chemical analysis, geological testing, modification of computer devices and other technological issues.

b) The key components of many expert systems are as follows and are illustrated as in figure 1.

i.A Knowledge Base

ii.An Inference Engine and

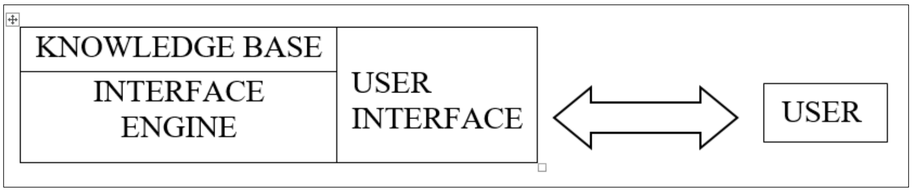

Figure 1. Schematic representation of an expert system

2.2 Knowledge Base

Performance of an Expert system relies on the use of as much domain knowledge as possible to give intelligent advice about a particular domain. The domain knowledge collected as the system's knowledge forms as a *knowledge base* which is the essential component of an expert system [1]. The general representation of a knowledge base is presented in figure 2.

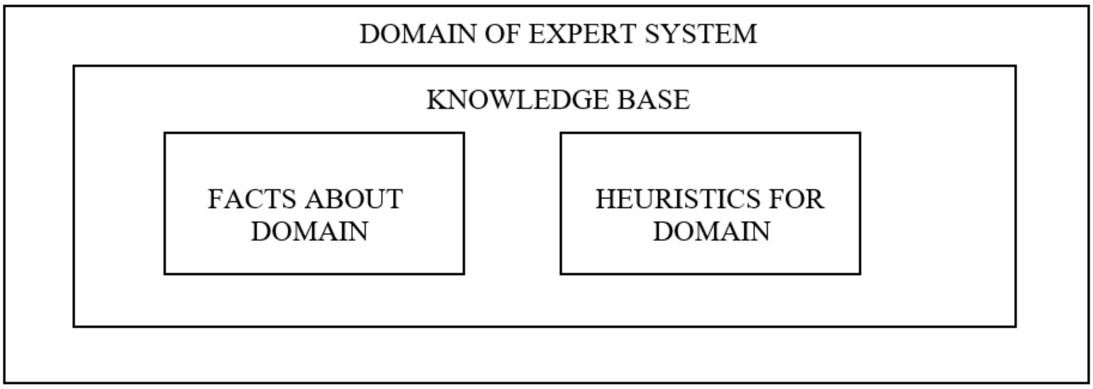

Figure 2. General representation of the knowledge base

The following are different ways of representing knowledge.

- Semantic Networks
- Object Attribute Value Triplets
- Rules
- Frames
- Logic Representation
- Blackboard Architecture

Each method has its own advantages and disadvantages. They are briefly explained as follows.

2.2.1 Semantic network

The semantic network is the most general representational scheme. A collection of objects called nodes can form a semantic network. The connections between the nodes are called arcs or links. Suitable labels are assigned to links and the nodes. However, no absolute constraints exist for naming nodes and links. A typical convention is presented as follows.

a. Nodes are utilized to represent object and descriptors.
   - *Objects* may be physical in form and seen as event, act, or abstract types.
   - *Descriptors* provided additional information about objects.

b. Links relate the object and descriptors. All types of relationships are represented using links. Following are the common links:
   - *Is-a* links are often used to represent the class/instance relationship.
   - *Has-a* links identify nodes that are properties of other nodes.
   - *Definitional links* provide a definition.

By providing additional routes, heuristic information gathering links will enrich the network. This representation scheme has a major Flexibility advantage. Another characteristic of the semantic network is inheritance, the ability of nodes to inherit the character of those related to it.

2.2.2 Object Attribute Value O-A-V triplets

Another common way of expressing truthful knowledge is through triplets as object-attribute-value. Within this scheme, target may be physical entities such as doors, or intangible entities such as bank loans or episodes of sales may be. Attribute describes the properties or attributes associated with the objects. Typical characteristics for a physical object are the scale, shape and colour. Tax is a Sales transaction attribute. In a given situation, the final triplet member is the value of an attribute. For example, the color of an apple may be red, or the interest rate on a bank loan may be 12 per cent. The O-A-V (Object-Attribute-Value) representation is a special case of semantic network. The object-attribute link is a “has-a” link, and the attribute value-link is an “is-a” link [5].

2.2.3 Rules

A knowledge base includes both declarative knowledge – data about things, events, circumstances and awareness of procedures – and information about action courses. Two types of knowledge may be separated or integrated, depending on the form of representation of knowledge chosen. The most prevalent form of representation of knowledge currently used is the approach to the rule-based system of production.

In a rule-based system, the procedural knowledge in the form of heuristic “if-then” production rules, is completely integrated with the declarative knowledge. Some production rules pertain to other production rules. These rules are known as meta rules.

Rules are used to represent relationships. Rules can be used with either A-V or O-A-V representation. Generally, the condition part is called a premise and conclusion part is called the conclusion. The statement in the premise block is called if-clause. The conclusion below then is called then clause. It is explained with a typical example as follows.

Rule:

Premise **If** number of protons is 16 and number of protons and neutrons are same.
Conclusion **Then** atomic number is 16.

Uncertain Rules can conclude values that are less than definite. The method of handling uncertain facts and relationship are undefined as explained in the following example.

Rule:

Premise **If** an animal is having a backbone
Conclusion **Then** the animal may be mammal.

Variable Rules: Some knowledge systems have rules which integrate "vector" pattern matching. The variable rule in these systems permits the program to insert several different facts into the same format. Below is an example of matching facts.

Rule1: **If** the tattoo is of a fish and the color of the fish scale is pink,
Then the origin of the tattoo is China.
Rule 2: **If** the tattoo is of snake and the color of the snake's scale is blue,
Then the origin of the tattoo is Hong Kong.
Rule 3: **If** the tattoo is of the dragon and the color of the dragon's scale is red,

Then the origin of the tattoo is Beijing. The key reasons for writing rule-based expert system is that it is generally intuitively appealing to human expert to communicate their subject experience in terms of situation-action pairs. Rules are a fairly standardized way of expressing information without complex constructs in programming. The knowledge must be encoded in a readily extensible and modifiable form. Also rule-based methods are well suited for representing procedural knowledge.

2.2.4 Frames

Frames offer a particular way of portraying evidence and relationships. A frame is a description of an object which contains slots for all of the object related information. Slots can store values, like the attributes. Slots may also contain default values, pointing to other frames, sets of rules, or procedures that may be used to get values. The implementation of these extra features distinguishes frames from O-A-V triplets. From one point of view, frames make for a richer representation of information while from the other they are more complex and difficult to create than simple O-A-V / rule systems.

2.2.5 Logical rpresentation

Logical expression provides another option for knowledge representation. Logic is concerned with truthfulness of chain of statements. Knowledge representation using logical expression is presented as follows.

- State (Tamil Nadu)
- City (Chennai, Bangalore)

The above representation shows that Tamil Nadu is a state, Chennai and Bangalore are cities.

2.2.6 Blackboard architecture

The Blackboard Architecture is a framework for facilitating a complex problem solving. The elements of a blackboard are entries, knowledge sources, blackboard and control mechanism. Blackboard is a hierarchical method of production that has three components: working memory, output memory and interpreter. Working memory contains besides other normal attributes, results of problem solving, and represents the current state of the problem-solving activity. It can be said to be equivalent of the human short-term memory. Production memory contains the problem-solving knowledge in the form of an unordered collection of production rules. Rules represent knowledge as condition-action or **if-then** pairs. It can be said to be equivalent to human long-term memory. Interpreter facilitates the communication between production memory and working memory. It performs the Match-Select-Execute cycle. Entries are intermediate results that are arrived at during the course of problem solving. These determine what actions will be taken next. Knowledge sources could contain procedures, rules, assertions, objects, etc. Knowledge does not communicate with each other directly. Communication is mediated by the blackboard. Knowledge sources are data or entry driven in nature and produce changes to the blackboard. Selection of the knowledge sources for execution is performed by the control mechanism. Blackboard has global database entries. Computation and solution states of the problem-solving activity are kept as entries. The blackboard can have multiple panes, organized hierarchically at various levels of abstraction. Entries can be grouped contextually. Contents of the blackboard change as the problem-solving activity continues. Control mechanism monitors changes in the blackboard and determines which knowledge source should operate. It uses selection criteria for selecting and scheduling. The blackboard architecture is represented in figure 3 [4].

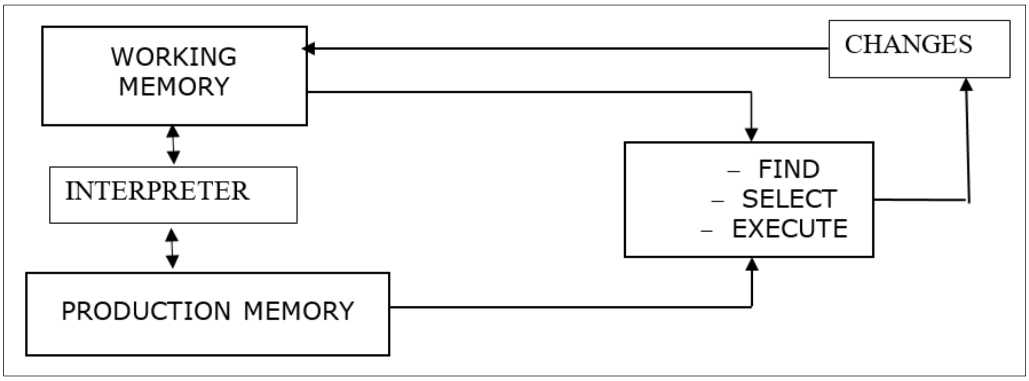

Figure 3. Blackboard architecture

2.3 Inference engine

Only getting the knowledge base now doesn't make an insightful expert computer. The program must have some other part that guides the information execution. This part of the system is recognized as control mechanism, rule interpreter or inference engine [1].

The inference engine determines which heuristic search techniques are to be employed for the rules of knowledge base. Then, these methods will be used to solve the problem. Essentially, an inference engine runs an expert system that determines the rules to be followed, provides access to the right rules in the knowledge base, implements the rules and decides when to find an appropriate solution. It refers to the process that navigates the base of information and addresses problems during a consultation. The inference engine is the intelligence "that helps the network of experts to draw conclusions based on the information or experience contained in the knowledge base" [1].

The two huge classes of inference engine are deterministic and probabilistic. The deterministic engine offers with the objects which hundred per cent fits with the supplied information whereas the probabilistic regularly is of the type of which a perfect match is impossible with the furnished facts [5]. This can be made clear with an assist of two examples. An atom can be recognized 100 per cent with its atomic number. This is a deterministic type. An example of probabilistic type is the sociological answers closer to the questions like “What is the first-class way to forestall, a pupil from shedding out of excessive school?” And answers will solely be a probabilistic one. Beyond the broad category of certainty and uncertainty [6], there are three basic approaches to construct an inference engine particularly Forward Chaining, Backward Chaining and Rule-Value.

2.4 User Interface

The user interface is defined as the part of a user-interacting expert system. The communication with the user interface is bidirectional. The customer has to be able to clarify the problem to the expert system, and the system has to be able to respond with its suggestions.

3. Design of Expert System study

3.1 System requirement

The design of structural systems is a branch in structural engineering that requires both the knowledge of the structure under the given loading conditions and also sound engineering sense so as to arrive at an economical solution from a few alternative design solutions point of view. Designing complex and large structures like ships and offshore structures require a great deal of expertise and experience on the part of the designer. Specifically, there is a need to develop suitable computer software for the design of steel plated local grillages subjected to dynamic loads [4]. An appropriate system development tool is needed for the design and creation of these software. Expert systems development tools such as ART (Automated Reasoning Tool), Expert Ease, EXSYS, KEE (Knowledge Engineering Environment), and KES (Knowledge Engineering System) are available. However, their full-scale practical implementation, however, will require high level expert system tools notably in the areas of Graphical User Interface and better explanation capability. Knowledge acquisition will require close coupling of data and other language interface. Already developed tools are specific to particular domain areas. Each tool has its own advantages and disadvantages. Thus, an effective shell is a pre-requisite before the knowledge bases are developed. This has led to the design and development of a general shell for the development of expert systems. The shell will have facilities for easy application of the information and for smooth interaction, possibly in graphical mode, between the user and the expert program.

3.2 Specification

Improvements and additional feature requirements have led to the need for an enhanced version of the development tool shell. Aims at improving many of the features and adding more features to downloading of a knowledge base is an important requirement since a program or a fragment of the program residing in the memory might tend to cause unexpected damage to other programs and the system as well. Comment line features form an important enhancement as they make a program understandable. This feature has to be made available to the user coding the knowledge base.

3.3 Software Environment

3.3.1 Introduction to windows programming. There is currently a stand-alone Windows operating system with the additional new features of thread-based multitasking and the console-based interface. The key features of Windows are as follows:

3.3.1.1 The windows call-based interface

Windows uses a *call-based interface* to access the operating system. To access operating system features the interface uses a set of system-defined functions. These functions are referred to as the Application Program Interface or the API. An application program calls the API function to communicate with Windows. These functions include all necessary operating system related activities, such as memory allocation, outputting to the screen, creating windows, and the like.

3.3.1.2 Dynamic Link Libraries (DLLs)

Dynamic Link Libraries or DLLs include the Windows API functions. Thus, each program compiled for Windows does not hold a copy of the API functions. Instead, each program has access to the DLLs when it is executed.

The functions are stored in a relocatable format within a DLL. When a program calls an API function during the compilation phase the linker does not add the codes for that function to the program's executable version. Rather, it adds instructions for loading the feature, such as what DLL it resides in and its name. When the program is executed, the Windows loader also loads the necessary routines. Thus, the API functions are added only when the application is loaded into memory for execution.

3.3.1.3 The Graphical Device Interface (GDI)

There is a subset to the API called the GDI (Graphical Device Interface), which is a part of Windows that provides device-independent graphics support. The GDI functions are what make it possible for a Window application to run on a variety of different hardware. Windows offers video and printer drivers to prevent the software from understanding the form of video board and printers that are connected to the device. Rather, the program calls GDI function which references a data structure called a system background to address the hardware. Windows maps the configuration of the system context to a physical unit and issues the correct input / output instructions. The GDI is as fast as direct access to video and allows to share the display with different applications written for Windows.

3.3.1.4 Multitasking in Windows

All Windows versions support multitasking. A process is an operating program. In multitasking based on method, two or more programs can run simultaneously.

A thread is an executable code unit which can be dispatched. In thread-based multitasking, individual threads within a single process are multitasked. All processes have at least one thread. Each active thread of execution is granted a slice of CPU time. When that time slice ends, execution automatically moves on to the next thread. This prevents a program from dominating the CPU [3].

3.3.2 The Visual C++ components

- *The Microsoft Developer Studio and the Build Process* - The Developer Studio is a Windows-hosted integrated development environment (IDE) that is shared by Visual C++, Microsoft FORTRAN, Visual Java and other products.
- *The Graphic Editor* – The Graphic Editor is used to edit all standard resources. It includes a WYSIWYG (What You See Is What You Get) menu editor, a dialog box editor and tools for editing icons, bitmaps, and strings.

3.4 Hardware environment

The hardware features used in the development of the Shell are as follows.

- High speed processor which follows from the usage of Visual C++.
- Large Memory – An expert system usually tends to be large and use more computer memory than other programs. They take memory on the RAM as well as the disk drive where they are stored. A minimum of 1GB RAM is a must.
- The GUIs used can be accessed either through the keyboard or by using a mouse.

4. Concluding remarks

Expert Systems is one of the important areas of Artificial Intelligence in problem solving. Knowledge Based Expert systems specifically enable the computer to assist in solving ill-structured and non-deterministic problems such as in design activities including synthesis, evaluation, and modeling and decision-making process [4]. Backward chaining method, Forward chaining method, Rule-Value method are the three major methods involved in solving these problems. These aspects differentiate Knowledge Based Expert System programming from normal programming. The development of an Expert System requires a tool that will provide for better means of knowledge acquisition, inference mechanism and user interface.

5. References

[1] Lake B M, Ullman T D, Tenenbaum J B and Gershman S J 2017 Building machines that learn and think like people *Behav. Brain Sci.* doi: 10.1017/S0140525X16001837.

[2] Abbott R 2020. Understanding Artificial Intelligence. *In The Reasonable Robot: Artificial Intelligence and the Law* (Cambridge: Cambridge University Press) pp. 18-35 doi:10.1017/9781108631761.002

[3] Davenport T, Guha A, Grewal D, and Bressgott T 2020 How artificial intelligence will change the future of marketing *J. Acad. Mark. Sci.* **48** 24–42 , doi: 10.1007/s11747-019-00696-0.

[4] Star S L 1989 The Structure of Ill-Structured Solutions: Boundary Objects and Heterogeneous Distributed Problem Solving *Distributed Artificial Intelligence* ed Les Gasser and Michael N. Huhns. (Amsterdam: Elsevier) chapter 2 pp 37-54 https://doi.org/10.1016/B978-1-55860-092-8.50006-X

[5] Reid L A and Polanyi M 1959 Personal Knowledge: Towards a Post-Critical Philosophy *Br. J. Educ. Stud.* ed Routledge and Kegan Paul (London: Taylor & Francis, Ltd.) Volume 8 No 1 pp 66-71 doi: 10.2307/3119338.

[6] Agrawal A, Gans J, and Goldfarb A 2019 Artificial Intelligence, Automation and Work, *The Economics of Artificial Intelligence* ed Agrawal A, Gans J, and Goldfarb A (Chicago and London: The University of Chicago Press) Chapter 2 No 8 p 197

[7] Ghahramani Z 2015 Probabilistic machine learning and artificial intelligence *Nature* **521** pp 452–459, doi: 10.1038/nature14541.