Issues in Structuring the Knowledge-base of Expert Systems

Eric C. Okafor[1] and Charles C. Osuagwu[2]
[1]Enugu State University of Science and Technology, Nigeria
[2]University of Nigeria, Nsukka, Nigeria
ericokafor@cee-esut-ng.org
cue@infoweb.com.ng

Abstract: The major bottlenecks in expert system development lie within the processes of eliciting and representing knowledge. Knowledge representation schemes combine data structures, and interpretative procedures that enable the extraction of the knowledge embedded in the data structures. A broad spectrum of knowledge types need to be represented, but available representation techniques are not optimum systems since they vary in level of expressiveness and power. Knowledge demands more than the conventional representation structures used for databases and information. This is because information is derived from processing, refining and analysing raw data. The extra refinement, analysis and addition of heuristics to information converts it to knowledge. This paper discusses the major issues in the quest for an efficient knowledge representation technique and assesses the performance and level of usefulness of some of the most successful approaches in knowledge representation.

Keywords: knowledge representation, knowledgebase, production rule, semantic nets, frames, propositional logic, predicate logic, fuzzy logic.

1. Introduction

An expert system uses a repository of human knowledge captured and stored in a computer to solve problems that ordinarily require human expertise. Its design focuses and then converges on a narrow area of expertise called a domain. Expert system is the end-point of knowledge engineering and the format used by the knowledge engineer to capture knowledge is called knowledge representation. A knowledge-based system usually provides consistent solutions, aids knowledge transfer to remote locations and training of non-experts. It can also handle uncertainty by making explicit the human knowledge used and ultimately performs better since it produces fewer errors than non-experts. Knowledge demands more than the conventional representation structures used for databases and information. This is because information is derived from processing, refining and analysing raw data and it is the extra refinement, analysis and addition of heuristics to information that converts it to knowledge. Different knowledge representation (KR) formalisms have emerged and the drive towards efficient KR has also led to the development of KR languages like PROLOG, KRYPTON (Brachman et al, 1983), and KL-1 (Woods, 1983). The choice of a KR technique is essentially dependent on a range of factors. By defining the characteristics of a given representation technique and determining their ability to handle the functional equivalence of a specific knowledge area, we can determine whether it is a suitable match (Jimes and Lucardie, 2003). In this paper, we build up current issues in structuring knowledgebases by first determining the features of an efficient KR system and factors that influence their expressiveness and power. Then, some KR techniques (rules, semantic networks, frames, logic) hitherto considered effective for structuring knowledgebases are presented and critically appraised to determine suitable application area(s).

2. Considerations in KR

The method of knowledge representation by an expert system affects the implementation, efficiency, speed and even maintenance of the system. To be considered as efficient, a KB structure must be capable of representing a broad spectrum of knowledge types categorised by Feigenbaum et al (1981-82) to include:

- Objects - information on physical objects and concepts
- Events - time-dependent actions and events that may indicate cause and effect relationships.
- Performance – procedure or process of performing tasks
- Meta-knowledge – knowledge about knowledge including its reliability, importance, performance evaluation of cognitive processors (experts)

Also, the use to which knowledge must be put is equally considered in KR. Firebaugh (1998) identifies three stages in the use of knowledge as:

- Acquisition – combining new knowledge with the existing KB system (Figure 1). Two levels of acquisition are identified: the low level structures existing as facts in the database and the high level structures that perform the 'learning' function of relating new to stored information.

Reference this paper as:
Okafor E C and Osuagwu C C (2007) "Issues in Structuring the Knowledge-base of Expert Systems" *The Electronic Journal of Knowledge Management* Volume 5 Issue 3, pp 313 - 322, available online at www.ejkm.com

- Retrieval – sorting through the KB for retrieval of facts and relationships to solve present tasks.
- Reasoning – knowledge is used at different stages for reasoning. Reasoning may be formal, procedural, by analogy, for generalisation and abstraction. Implementation of deductive reasoning in KB systems is far easier than inductive reasoning (implied by the last three). Abductive reasoning is used when a complete set of information is not available to enable deduction, but this handicap does not restrict or deter decision-making.

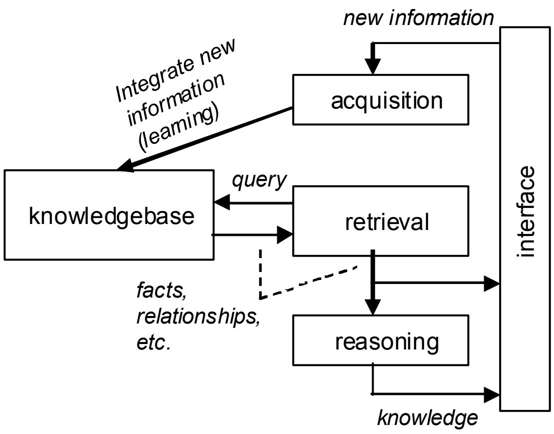

Figure 1. Illustrating the stages in use of knowledge

Summarising, a good KR system should posses the ability to represent the required knowledge, produce knew knowledge by manipulating represented knowledge, store and provide productive guides that direct the inferential mechanisms, and use automated tools to acquire new knowledge where possible. Most of the currently available systems have failed to optimise all these properties. The suitability of a KR system for certain applications may also be assessed based on certain other features. Firebaugh (1998) defines scope of a KR system as a measure of the domain from which knowledge is represented. The level of detailing of knowledge possible from a KR system is called the grain size and must be adapted to the details of the object represented in order to maximise efficiency. Modularity, a highly desirable feature, ensures that procedures or production rules are independent and may be modified and/or substituted without changing the structure of the system. Thirdly, the representation of tacit (implicit) knowledge requires certain considerations different from explicit knowledge. Tacit knowledge is logically embedded while a system usually enjoys direct access and can manipulate explicit knowledge. Tacit representations are efficient mostly in terms of memory usage. Finally, where knowledge is procedural (encoded in functions/procedures), its representation is easily accomplished by use of conventional computer languages (example PASCAL). For declarative knowledge, use of predicate logic or languages of declarative nature (example PROLOG) is advised.

3. Power of a KR system

Certain indices serve well when used to measure the power of a KR system. Ability to Specify Object Properties and Values – The ability to provide means of indicating property and property values of an object is considered a measure of the expressiveness of a KR system. Ability to Express Hierarchical Structures – This provides valuable tool for refining knowledge. An advantage of such a hierarchical taxonomy is the concept of inheritance – subclasses automatically acquire the characteristics of classes up the hierarchy. Ability to Extract Meaning (Semantics) – One of the major problems of natural language processing (NLP) is determining the semantics of a given communication. This meaningful content is the knowledge. An efficient KR system must provide for association of meaning with information represented. Firebaugh (1998) again posits that achieving this demands the following features from a KR system:

- Support for truth theory – provide measurable indices for propositions or specifying what propositions mean
- Provision for constraint satisfaction – clearly specify rules, boundary conditions (otherwise constraints) to solving a problem
- Support for incompleteness and fuzziness – demands that schemes for representing fuzzy knowledge and harvesting appreciable meaning from the stored knowledge be provided

- Provide for common-sense reasoning – these are schemes that enable the representation of transparently obvious knowledge.

4. KR structures

4.1 Production rule

4.1.1 Features

This is a knowledgebase structure that models the elicited expert's knowledge into a set of "situation-action" rules as presented below.

IF in *situation* type1
THEN perform *action* for type1
AND IF in *situation* type2
THEN perform *action* for type2
. .
etc.

These rules are known as IF-THEN rules, production rules, or situation-action rules. Rules are called production rules because new information is produced when the rule fires. Production rule is appropriately used when there is a chain of knowledge. Rules are linked into chains of reasoning by the expert system, which can use either backward chaining or forward chaining. In other words, knowledge fired into the system concatenates until a solution is reached. This method is similar to the way in which experts formulate knowledge in a "cause-and-effect" fashion. It is useful when representing heuristics (rule of thumb) of how the expert does things or what he does and does not know why (implicit knowledge). Five major properties of production rules as outlined by Hayes-Roth (1985) are:

- Incorporating human skills in conditional IF-THEN rules
- An increase in skill proportional to the enlargement of the KB
- Power to provide solutions to complex problems by selecting rules and combining the result
- Adaptive determination of the most suitable sequence of rules to execute
- Providing clear insight into the result by reversal of the line of reasoning

4.1.2 Advantages of using production rules

Production rules have the following advantages when used to represent knowledge in the knowledgebase.
Naturalness of expression - The "cause-effect" or "situation-action" format of production rule models the expert's knowledge in a natural way much like the way the expert himself reasons, thereby making knowledge representation easier. Modularity - The use of production rules to represent the expert's knowledge enforces modularity since each rule stands as a module on its own. As a result KB structures using production rules are very easy to debug, modify and maintain.

4.1.3 *Disadvantages of using production rules*

Inefficiency - This is because there are cases where modularity and prioritisation of rules alone cannot solve the problem of rule dependency. For instance, a situation where "Rule A" has preference over "Rule B" and "Rule B" has preference over "Rule C", but "Rule C" has preference over "Rule A". Inexpressiveness – KBs structured using production rules are not very expressive since the rules are simply rules of thumb, chunks of knowledge (heuristics) on how to do something. In other words, it is difficult to represent tacit (unconscious) knowledge of the expert using production rules.

4.2 Semantic network

4.2.1 Features

Semantic network (or propositional network) is a KB structuring format that represents knowledge (objects, concepts, situations and actions) as nodes with labeled arcs representing the relationship among them. The labels in the nodes gain their meaning from their connections (Brachman, 1979). Semantic network was developed first as a way of representing human associative memory and language understanding (Quillian, 1968). Relationships are of paramount significance in semantic network because they provide the basic structure for organising knowledge. Without relationships, knowledge is simply a collection of unrelated facts. With relationships, knowledge is a cohesive structure about which other knowledge can be inferred (Giarratanno, 1998). Although there are no constraints in the naming of nodes and arcs, *IS_A* and A_KIND_OF usually annotate arcs to depict the relationships between objects, attributes, situation or actions. Consider a segment of the knowledgebase for an expert system (CONFIG_EXPERT) used to analyse and configure a personal computer (PC) system and based on the following elicited knowledge from a human expert.

Kn1: Drives are components of a computer system.
Kn2: HDD, FDD, CDD, DVD, FLASH are kinds of Drives.
Kn3: SCSI and IDE are kinds of HDD.
Kn4: FDD and HDD operate based on the principle of magnetism.
Kn5: DVD and CDD are optical drives.

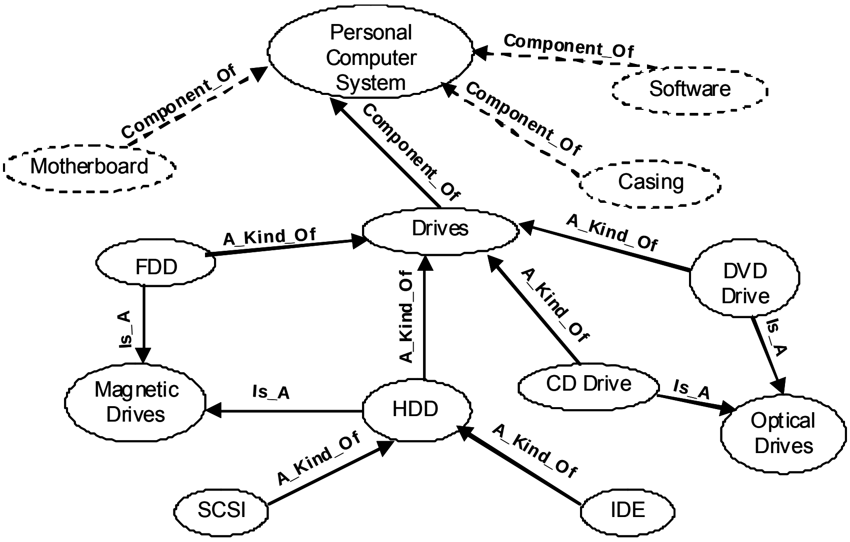

Figure 2. Semantic network structure for CONFIG_EXPERT.

Figure 2 shows an acceptable structuring of this knowledge using semantic nets. A network that exhibits property inheritance is easily implemented by combining the lattice structure defined using IS-A links with the attribute links. In such a structure, objects at the lower level (sub-class) inherit properties explicitly associated with objects at the higher level (class). Then, more general classes mostly appear at the top while specific ones (instances) tend to the bottom of the structure. Semantic networks like this can be encoded as a series of computer-readable statements. Niemann, et al (1990) presented the development of a system for the representation of knowledge using semantic nets and demonstrated how it can be used for knowledge-based image and speech understanding.

4.2.2 Problems of semantic network

Lack of theory of semantics - Knowledge is not definitively represented because of the semantics that may result from naming nodes and lack of fundamental distinction between types of links. For instance, little or no difference exists between links that describe objects and those that make assertions. The same is true for links between classes of objects and instances or individuals (Woods, 1975). Combinatorial explosion of searching nodes - The possibility of combinatorial explosion exists since a search that will return negative result may have to trace all links in a net. Logic and heuristic inadequacies - They can neither define knowledge the way logic does nor embed heuristic information on how to efficiently search the network.

Embedding procedures (in a node) that are automatically invoked when the node becomes active has been tried as heuristic enhancement. However, heuristic and logic enhancements provided little gain in system performance especially if one must maintain the natural expressiveness of semantic nets (Giarantano, 1998).

4.3 Frame systems

4.3.1 Features

The concept of frames grew out of the need to represent knowledge in the context of which many objects and events appear. Minsky (1975) describes frames as a network of nodes and relations with the top level as attributes and the low levels as terminals or slots filled by specific instances or data. Semantic nets represent knowledge in two dimensions, but frames extend this by providing for nodes to have structures that vary from simple values to other frames. Essentially, they are best suited for representing 'stereotype' situations, objects and events especially about a narrow subject possessing default values. The attraction to frames is its representation of knowledge, which is easier to comprehend than logic or production systems (Jackson, 1986).

"Integrated-Circuit_Classification"	
Slot_Name	**Fillers**
Specialization_Of	A_Kind_Of Electronic component
IC_Technology	String (Default→ CMOS) ***If_Needed***: Procedure ***FIND_TECH***
Number_Pins	Integer (Default→3) ***If_Needed***: Procedure ***FIND_PIN***
Type_Of_Signal	(Analog, Digital, Hybrid) ***If_Added***: Procedure ***ADD_SIGNAL***
1.1 Integration_ Level	(SSI, MSI, LSI, VLSI)
Package _Style	(DIP, SIP, PGA, FLATPACK, TIP, SMT, LCC)

Figure 3. Frame for classification of integrated circuits.

Individual frames, corresponding to nodes of a semantic network, store information about object and class of objects. The arcs of a semantic net are here represented as relationships between frames. Referred to as object-attribute-value, a frame is typically made up of a frame name (the object), slots name (attributes) and the fillers (values of each slot). It can be thought of as a record data structure consisting of a number of "slots" and associated with each slot is a "value". The slots form a description of the object, with each slot-value pair corresponding to a common attribute of the object.

In a KB, frame slots may hold information such as rules, graphics, comments, question for users, another frame or hypothesis concerning a situation. The slots may also feature procedural attachment that may take any of the following forms (Figure 3):

If_Needed procedure - executed when a filler value is needed but none is initially present or the default value is not suitable. If_Needed procedures FIND_TECH and FIND_PIN are executed in Figure 3 when the technology of the IC is not CMOS and number of pins not 3 respectively.

If_Added procedure - executed when a value is to be added to a slot. In Figure 3, the If_Added procedure ADD_SIGNAL allows the inclusion of other signal types outside those listed.

If_Removal procedure - executed whenever a value is to be removed from a slot; probably if the value is obsolete.

In practice, no slot value is ever left empty within a frame; rather it is filled with a default value inherited from the "ancestors". These default values form the stereotypical object, and are overwritten by values that better fit the more specific case. In deducing the frame for a personal computer (PC), a type of computer, one expects sub-frames for drives, microprocessor, memory, software, motherboard, et cetera (see Figure 4). Refinements may develop other sub-frames for the objects listed. For example, drives may contain other sub-frames for magnetic disk drives, magnetic tape drives, optical drives, flash drives, et cetera.

4.3.2 Daemons

Daemons are procedures activated anytime during program execution depending on conditions evaluated in the daemon. Examples of daemons used in conventional programming include error detection, default commands and end-of-file (*eof*) detection. Daemons are included or attached to slots in a frame system to improve efficiency. IF_NEEDED and IF_ADDED procedures described previously may be used to provide daemons to frames.

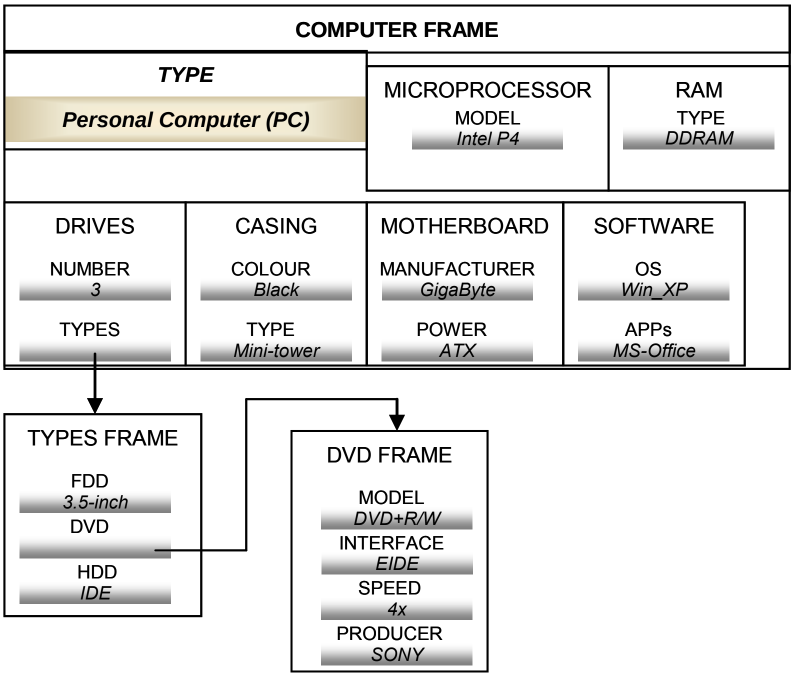

Figure 4. Representation of a PC illustrating the hierarchical structure of frame system.

4.3.3 Advantages of frames

- When used to structure KB, frame system represents knowledge much like they appear ordinarily. This ‘stereotypical’ structure assists both humans and machines in understanding.
- Also, a number of different objects may share the same frame; hence, it assists in data compression.
- Frames provide a natural hierarchy and inheritance of properties through the sub-frame structure, thereby making knowledge representation easier.
- Frames support default reasoning because of the ability of a slot to inherit default value They can be used to tackle the problem of incomplete knowledge when building the knowledgebase.
- The structure of frames supports smooth system query since each frame contains all related information in the slot.
- Finally, they support procedural attachment (including daemons), which could be used to provide more information to slots, hence allowing the system to adjust to new situations.

4.3.4 Disadvantages of frames

- Frames lack semantics. In other words, there is no standard definition of links used to depict generic frames from specific frames. The IS_A and A_KIND_OF relations may be confused for one another.
- Since default values may be overwritten, one crucial type of representation becomes impossible – that of composite description whose meanings are functions of the structure and interrelation of their parts (Brachman, 1985).
- Overwriting default values may become very problematic because of inheritance feature down the hierarchy.

4.4 Logic

Logic is a way of declaratively representing knowledge. Logic-based knowledge representation hinges on syllogism, which is made up of two premises and a conclusion that is inferred from the premises. For example

Premise:	Microprocessor is built using semiconductor
Premise:	Intel Pentium 4 is a microprocessor
Conclusion:	Intel Pentium 4 is built using semiconductor

The application of logic to knowledge representation and exact reasoning in computers resulted in logic programming and the development of logic-based languages like PROLOG (PROgramming in LOGic). The use of logic for knowledge representation may be classified under three headings depending on the certainty of the knowledge statement being represented in the knowledgebase.

4.4.1 Propositional logic

This KR scheme is a symbolic logic for manipulating propositions represented as logic variables. Propositional logic is used to represent elicited expert knowledge, which has finite certainty truth-value. For instance, the following elicited knowledge can be structured using propositional logic, since each has a definite truth-value.

Kn1: Virus damages computers.
Kn2: PC-cillin removes virus

Propositional logic uses logic variables to represent elicited knowledge statements. These logic variables can subsequently be combined in any format using logical connectives such as:

^	for	AND
v	for	OR
~	for	NOT
⊃ or →	for	IF….THEN ….
↔	for	IF AND ONLY IF

With the combinations, complex and compound knowledge statements can be represented. For example, consider a segment of the knowledgebase for our expert system on configuring PCs:

Kn: *I*f PC has virus THEN use PC-cillin.

This can be represented in the form:

$$P \rightarrow Q$$

Where P = PC has virus
Q = use PC-cillin.

4.4.2 First order predicate logic

The first order predicate logic was introduced to handle the major problem of propositional logic - the inability to represent incomplete statements. Predicate logic is the basis of logic programming languages such as PROLOG. In predicate logic, special words called *quantifiers*, such as “all”, “some”, and “no” are used to analyse the internal structure of knowledge statements. All quantifiers are concerned with “how many”, “to what extent” or “to what degree” is the knowledge statement valid, thus permitting a wider scope for expressing uncertain or incomplete knowledge. Some of the quantifiers used in predicate calculus or logic are:

∀	Which means	“For all”
∃	Which means	“For some”.

Predicate logic helps describe the properties of knowledge subjects or situations in a knowledgebase with the help of *predicate functions*. For example, consider the following knowledge statements:

Kn1: Some computer viruses cause software malfunction
Kn2: PC-cillin is the anti-virus software for all computer viruses

Which in predicate calculus is:

$\exists$ x,y: Cause Of (x,y)
$\forall$ z,x: Antivirus For (z,x).

Where logic variable x represents computer viruses, y represents software malfunction, and z represents PC-cillin. Coding in a logic programming language (like PROLOG) may proceed thus:

Cause_Of (Some_Viruses, Software_Malfunction)
Antivirus_For (PC-cillin, Computer_Virus)

4.4.3 Fuzzy logic

Fuzzy logic (Zadeh, 1978) is designed for the representation of knowledge of partial or incomplete truth-value. It is a nice tool for handling the problem of elicited expert knowledge with uncertain truth-value. Consider, the following elicited knowledge from an electronic engineer:

"High current transistors are classified as power transistors."

Element of uncertainty is introduced in trying to fix values of current that may be classified as 'high'. Membership function in fuzzy logic helps define the degree of membership of transistors having a certain range of current rating (here assumed from 100mA to 800mA) in the class of power transistors. A typical representation of such membership function *μ(x)* is

$$\mu(x) = \begin{cases} 0 & \text{if current}(x) < 100mA \\ (\text{height}(x) - 100mA)/100mA, & \text{if } 100mA <= \text{current}(x) <= 800mA. \\ 1 & \text{if height}(x) > 800mA. \end{cases}$$

Membership value of '0' indicates complete non-membership of that range of transistors to the class of power transistors while '1' indicates complete membership. However, the resulting fractional values between 0 and 1 define the degree of membership of each transistor in the class of power transistors.

4.4.4 Problems of logic-based structures

- With logic only, it is difficult to express procedural knowledge in the knowledgebase. Also, logic-based KB structure does not support default reasoning, which is an essential feature of a good KR scheme.
- The output of a fuzzy rule-based system is generally imprecise and fuzzy. Defuzzification methods (weighted average, centroid, max-membership, et cetera) are required to convert fuzzy conclusions into one precise quantity (Devaraj, et al, 1999)

5. Example of KB structure

Figure 5 shows a section of the structure of the KB for CONFIG_EXPERT, an expert system used to configure personal computer systems. The overall system aims at providing expert advice on how to configure the system unit of a personal computer from basic components as motherboard, drives, casing, software, et cetera. The illustration is fairly simply and limited to drives. Popular PC drives are magnetic (floppy disk - FDD, hard disk – HDD), optical (compact disk – CDD, digital versatile disk – DVD) and flash (semiconductor mass storage devices). Different technical considerations apply when configuring each drive for use in a PC. To achieve the desired effect, combinations of various KR schemes are used in most designs. In the example, nodes of semantic network hold objects as well as their attributes while COMPONENT_OF, A_KIND_OF, IS_A are used to depict relationships. Rules presented in "cause-effect" manner are used to represent the rules of thumb (heuristics) elicited from the human expert. This essentially ensures well-grained knowledge representation.

6. Conclusion

Knowledge representation is of major importance in expert systems development. Available representation schemes vary in level of expressiveness and power. This paper has presented the considerations in the choice of a representation scheme and the indices that may be used to measure the power and expressiveness of a KR system. Each KB structuring scheme has merits and demerits. For greater efficiency and better knowledge representation, a combination of two or more may be needed for structuring a particular knowledgebase. In such a situation, each scheme deploys to structure part of the knowledgebase it suits best. An example in the paper illustrates this concept.

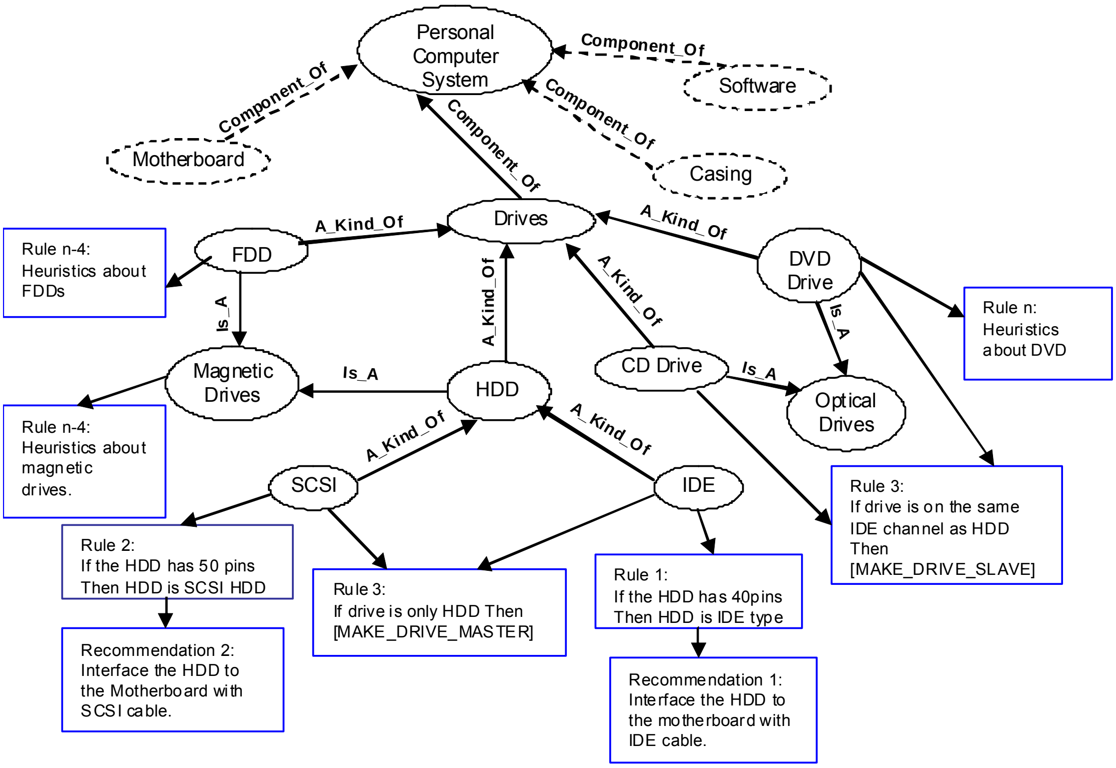

Figure 5. Illustrating section of the knowledgebase structure for CONFIG_EXPERT.

The forms, patterns, as well as structures that elicited expert knowledge take in the knowledgebase are of paramount importance since they determine the efficiency. The most common structures - production rules, semantic networks, frame systems and logic (predicate calculus and fuzzy logic) - are discussed and their merits and demerits outlined. Just as other tools, each should be applied for the purpose it best serves, the prime focus being on creating well-grained and efficient KR. The future of KR seems to be focused on providing greater visual component. The fusion of today's graphical display and graphical input device technologies is facilitating the development of visual as well as textual representation of knowledge (Messinger, et al, 1991, Crowther and Hartnett, 1997, Chen, et al, 2001). Such pictorial representations of KB structures promise to greatly enhance the pictorial modelling and development of expert systems that provide for visual conception and communication of abstract information.

References

Brachman, R. J. (1979). On the epistemological status of semantic networks. In Findler, N. V., editor, *Associative Networks: The representation and Understanding in Computers*, pp 3-50. New York: Academic Press Inc.

Brachman, R. J., Fikes, R. E. and Levesque, H. L. (1983). KRYPTON: A functional approach to knowledge representation. *IEEE Computer*, 16, pp. 67.

Brachman, R. J. (1985). 'I lied about trees'. *AI Magazine*, vol. 6, pp 80-93.

Chen, C., Kuljis, J., Paul, R. J. (2001). Visualising latent domain knowledge. *IEEE Trans, Systems, Man and Cybernetics – Part C: Applications and Reviews*, 31(4), pp. 518-529.

Crowther, P., Hartnett, J. (1997). "Eliciting knowledge with visualisation – instant gratification for the expert image classifier who wants to show rather than tell", Proc. of Annual Conf. of Geo-Computation '97 and SIRC '97. Univ. of Otago, NZ, pp 15-23.

Devaraj, D., Murthy, T. V. S. L. N., Yegnanarayana, B. (1999). "A fuzzy system model for plant condition monitoring", *Proc. Intern. Conf. Computer Modeling, Simulation and Communication (CMSC '99)*. New Delhi: Tata McGraw-Hill, pp. 210-214.

Fiegenbaum, E. A., Barr A., and Cohen, P. R. (eds) (1981-1982). *Handbook of Artificial Intelligence*, Vol. 1-3, HeurisTech Press, William Kaufmann Inc., Stanford, CA,.

Firebaugh Morris W. (1998). *Artificial Intelligence: A Knowledge-Based Approach*, PWS-KENT Publishing Company, Boston, pp. 275-303

Giarratano, Joseph C. (1998). *Expert Systems: Principles and Programming*, PWS-KENT Publishing Company, Boston. pp.63-106
Haye-Roth, B. (1985). “A blackboard architecture for control”, *Artificial Intelligence*, 26, pp251-321.
Jackson, Peter (1986). *Introduction to Expert Systems*, Addison-Wesley.
Jimes C. and Lucardie L. (2003). “Reconsidering the tacit-explicit distinction - A move towards functional (tacit) knowledge management”, *Electronic Journal of Knowledge Management*, Vol. 1, issue 1, pp 23-32.
Messinger, E. B., Rowe, L. A., henry, R. R. (1991). “A divide-and-conquer algorithm for the layout of large directed graphs”, *IEEE Trans. Systems, Man and Cybernetics*, 12(1), pp 1-11.
Minsky, M. (1975). “A framework for representing knowledge”, In *The Psychology of Computer Vision* Winston Patrick (editor),. McGraw-Hill, pp 211-217.
Niemann H. et al (1990). “A semantic network system for pattern understanding”, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(9): Pgs 883-905.
Quillian, M. R. (1968). “Semantic memory”, in *Semantic Information Processing*, Minsky M. M.(Ed), M.I.T. Press.
Woods. W. A. (1975). “What’s in a link: foundations of semantic networks?” in *Representation and Understanding: studies in cognitive science*, Bobrow, D. G. and Collins, A. M. (Eds), , Academic Press, NY, pp. 35-82
Woods. W. A. (1983). “What’s important about knowledge representation?”, *IEEE Computer*, 16, pp. 22
Zadeh, L. A. (1978). “Fuzzy sets as a basis of theory of possibilities”. *Fuzzy Sets and Systems*, 1(1), pp. 3-28.

