Smart Computing Review

A Survey on State-of-the-Art Knowledge-based System Development and Issues

Muhammad Muneer Umar[1], Amjad Mehmood[1], and Houbing Song[2]*

[1]Institute of Information Technology, Kohat University of Science & Technology / Kohat, KPK, Pakistan / muneerumar@hotmail.com, amjadiitkust@gmail.com

[2]Department of Electrical and Computer Engineering, West Virginia University / Montgomery, WV 25136 USA / Houbing.Song@mail.wvu.edu

*Corresponding Author: Dr. Amjad Mehmood

Received October 3, 2015; Revised November 20, 2015; Accepted December 15, 2015; Published December 31, 2015

Abstract: The human brain can store countless folds of knowledge. Still, we cannot fully utilize a single human brain to solve a specific problem. Knowledge-based systems (KBSs) are computer programs specifically developed to perform problem solving like human experts. These systems effectively expand efficiency and flawlessly solve problems in various fields. The structure of a KBS can be divided into five standard components, which are described in detail. The component used for storage is called a knowledgebase, while an inference engine is a software module that processes the knowledge stored in the knowledgebase. Different tools, shells and programming languages can be used to develop and utilize KBSs. The most popular development languages are List Programming (Lisp), Prolog, Java Expert Systems Shell (JESS) and C Language Integrated Production System (CLIPS). This paper highlights various known issues in the phases of the development, deployment and maintenance of KBSs. Developers, as well as end users, may face various issues in the life cycle of these systems. The key personalities involved in the development and maintenance of KBSs are knowledge engineers and domain knowledge experts. Most issues are related to the experts' availability, their behavior and long-term cooperation.

Keywords: Knowledge, Knowledgebase, KBS, Inference Engine, Expert Systems

DOI: 10.6029/smartcr.2015.06.001

Introduction

The lexical meaning of knowledge in the Oxford Dictionary is "facts, information, and skills acquired through experience or education; the theoretical or practical understanding of a subject." According to Fang et al. [1], we can classify knowledge in various ways, such as domain knowledge, explicit knowledge, tacit knowledge, heuristic knowledge and common sense.

Knowledge-based systems (KBSs) are rule-based systems formed as computer applications that collect knowledge from experts, either human or artificial, then symbolize and store it for solutions to complex or similar problems that they experience in the future. These systems perform reasoning and use different inputs to address unsolved hitches. The common element that unites all KBSs is an effort to represent knowledge explicitly, via tools rather than implicitly via code, as a normal program does [2]. Tools can be ontologies and rules applied to facts. These systems are useful for replacing the need for human experts to tackle problems in medical diagnosis, manufacturing, staff training, process monitoring, etc.

KBSs were first developed by artificial intelligence (AI) researchers. These are most of the time referred to as expert systems (ESs). However, we can separate KBSs from ESs [3]. An ES is used to solve a problem with the aid of a human expert, whereas KBS refers to the architecture of a system that it represents explicitly, rather than as procedural code. We can say that all ESs are KBSs, but the reverse is not true. Both systems use a provided or stored set of rules for handling complex problems. An ES depends on having enough information to address the accomplishment of different tasks, whereas a KBS collects a smaller amount of human knowledge and then reasons through a problem to discover a solution. The first KBSs were rule-based ESs.

One of the most famous KBSs was MYCIN, a program for medical diagnosis. The real-world facts were represented as a simple assertion stored in a flat database. There were some rules defined to manipulate the facts. Such systems, which represent knowledge explicitly via rules, have the following advantages.

- **Acquisition and maintenance:** There is no need for a programmer to maintain the facts. The domain experts can define and maintain the rules themselves.
- **Explaining:** Existing facts can be used to infer a new conclusion, and results can be explained for usage purposes. A simple example can be to follow a series of inferences towards a diagnosis, and then use these facts to clarify the diagnosis.
- **Reasoning:** Decoupling knowledge from the processing of that knowledge enabled general-purpose inference engines to be developed. That system can bring new solutions never seen before by developers, which follow from a data set stored in the knowledgebase [4].

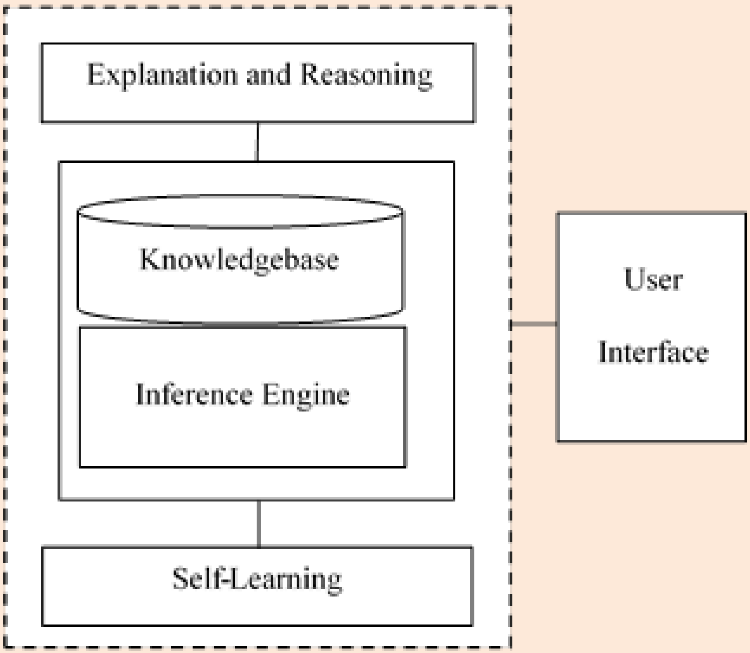

Figure 1. Structure of a Knowledge-based System

The five principle components of these systems (the knowledgebase, the inference engine (IE), explanation and reasoning, the user interface and self-learning) are discussed in the following sub-sections.

■ Knowledgebase

A KBS uses its knowledgebase to store all the knowledge data. The gathered knowledge from experts, either human or artificial, is first symbolized and then stored. The knowledgebase cannot be considered a typical database [5]. It does not operate on tables, which store strings, numbers and sometimes other objects; rather, it drives the pointers to objects that use further pointers. A knowledgebase can be represented as an object model, with classes having subclasses and instances. This can also be called an ontology, as in AI. The form of data stored in the knowledgebase is structural. Examples are DBLP, Google Scholar, YAGO, DBPedia, Wolfram Alpha and Freebase.

The KBS symbolizes human knowledge in the form of a coded body of a mathematical model. In the context of development of a KBS, we use two major terms: (a) knowledge engineering, which refers to development, and (b) elicitation, or the task of gathering and representing human knowledge [6]. The most popular mathematical formalism for KBSs is the Bayesian network (BN) [7], which represents knowledge through conditional probabilities and acyclic graphs. A BN is based on a set of variables and on a directed graph. In the graph, the nodes have one-to-one correspondence, and there is probabilistic independence relations among the variables based on a Markov condition [8].

■ Inference Engine

The IE is an important component of KBSs. It is software that works on the knowledgebase. It applies logical rules to the facts stored in the knowledgebase to deduce new knowledge. The IE works on either forward chaining or backward chaining. Forward chaining asserts new knowledge from stored knowledge, whereas backward chaining starts with solutions and decides what knowledge must be stated, so that the problem can be solved. An IE operates on the typical structure of the IF-THEN rule. In other words, we can say that an IE uses syllogisms for the deduction process, in which we use some propositions (known as premises) to determine a resulting proposition, called the conclusion.

An IE operates in three states: i.e. matching rules, selection rules and execution rules [9]. The IE filters all the content of the data that are satisfied by the rules. The content items are considered execution candidates. In the selection rules (the second state of the IE), some selection strategies are applied to further filter the rules for execution. In the broader AI context, the term heuristics is used for these selection strategies. Finally, the data are executed in the third state of the IE, where the right-hand sides of the stored rules are changed, or further processed, by any input from outside the IE, either from the user through user interface (UI) or by a function or program call. The IE is a finite state machine, and it cycles through its three states each time. We can structure an IE with the following diagram [10]:

Some of the known inference engines are described in the following.

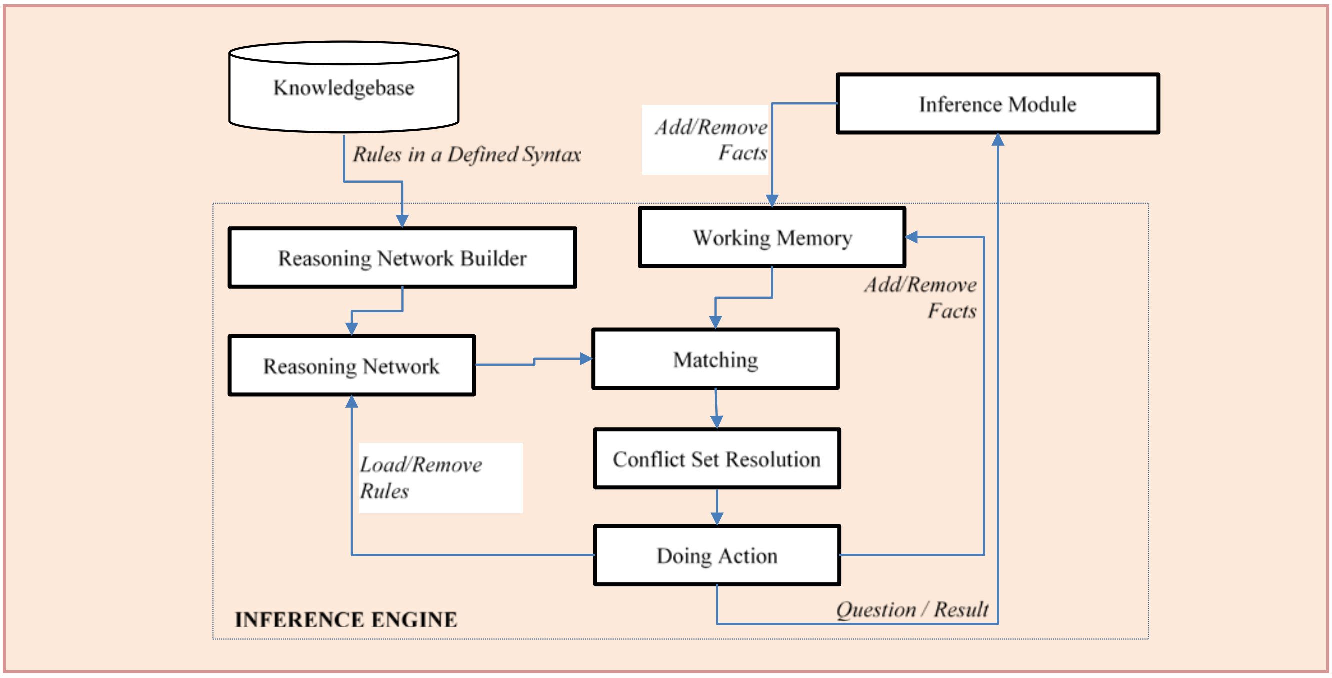

Figure 2. Typical Structure of the Inference Engine

- BaseVISor is a multipurpose forward-chaining IE. It is a Java application and API. It needs JRE 1.5 for execution. Its main specialty is that it takes data facts in resource description framework (RDF) triples format with an added provision for XML Schema data types and Web Ontology Rule Language (OWL 2 RL) [11].
- Web-Scale Parallel Inference Engine (WebPIE) is a MapReduce distributed RDFS/OWL. It reduces the operations by preparing a set of Map and Reduce operations by encoding semantic web reasoning. It uses the power of the MapReduce programming model for simplified administration, efficiency and error-tolerance. It is very scalable in terms of data size and in the number of processing nodes. This is accomplished by improving the execution of joins obligatory to apply the rules of RDFS and OWL-Horst [12].
- Variational Inference for Bayesian Networks (VIBES) uses variational methods for learning a wide range of probabilistic models. It solves problems variationally, short of options for coding. VIBES specifies models graphically. It simply constructs the model's Bayesian network by drawing a network into a graph and assigns properties to all the nodes [13].
- CR-MODEL is an IE for CR-Prolog and is sufficiently capable of permitting practical practice of the language for medium-sized applications. In case we have two rules, r1 and r2, then CR-Prolog forms it as prefer(r1,r2). The exclusivity property is used, which means that if r1 is preferred over r2, then the answer set will not contain the both r1 and r2. CR-MODEL uses this preference property [14].
- Aesthetic Quality Inference Engine (ACQUINE) automatically determines and then rates the aesthetic value of images uploaded by users in real time. It is used on the website photo.net for this purpose. It studies visual factors, like color distribution and photo composition, to determine a rating from 0 to 100 [15].
- OWLIM is an efficient semantic repository developed using the Java language. The Sesame RDF database uses it as a storage and inference layer (SAIL). OWLIM is based on the Triple Reasoning and Rule Entailment Engine (TRREE). TRREE is an innate RDF rule-entailment engine. The semantics supported can be formed through rule-set definition and selection. The most expressive pre-defined rule-set pools unconstrained RDFS with most of OWL Lite [16].
- F-OWL is implemented in Flora-2 and Extended Stony Brook (XSB). It is a frame-based system that is used to reason with the OWL language based on F-Logic ontologies. Flora-2 is a compiler that piles up from a dialect of Frame logic into XSB. XSB, developed by Stony Brook University, is a logic programming system having some additional features, like tabling. F-OWL reasons with OWL ontologies by using a simple importer, which takes RDF triples from a uniform resource identifier (URI) [17].
- Fast Classification of Terminologies (FaCT) is a description logic (DL) classifier that also works for testing modal logic satisfiability. The logic, in this system, is denoted by S. It comprises two reasoners: first for the logic SHF, and second for the logic SHIQ. SHF adds the logic S with a hierarchy of roles and functional roles. While SHIQ offers an addition of inverse roles and fully qualified number restrictions. Both the semantic reasoners use tableaux algorithms. [18]
- FaCT++ is the next generation of FaCT. It has an enhanced internal architecture with the same algorithms of FaCT. Moreover, FaCT++ is implemented using C++. Therefore, it is a more portable and efficient tool [19].

Explanation and Reasoning

A KBS relies on the knowledge gathered from experts. This can be done by reasoning with the experts and creating sessions of two-way explanations with them [20]. Sometimes, human experts learn new, or remember forgotten, things from the KBS. A typical KBS must provide an explanatory facility. There should be a mechanism for explaining the inside information and the reasoning process. Such a facility assists users, as well as experts, in the manipulation of knowledge.

A common reasoning algorithm consists of (a) creation of a conflict set, the appropriate rules, by applying conditions, and (b) selection of the most apposite rule from the conflict set; and then conflict resolution strategies are used for (c) execution of the rule for the proof or creation of a new rule, and (d) termination of the algorithm or another cycle of the whole process.

KBS or ES reasoning comes in two types: (a) induction, or backward chaining, and (b) deduction, or forward chaining. According to Liu et al. [21], reasoning can be differentiated into the following five technologies: rule-based reasoning, narrative-based reasoning, ontology-based reasoning, case-based reasoning and genetic algorithms.

User Interface

This is considered the most vital part of a KBS, because it provides bi-directional communication between user and system. Most of the KBS systems are shells based on a reasoning architecture. Some interfaces are graphical and provide object-oriented programming (OOP) as a language and hypertext (along with inductive reasoning) for solving problems through examples. Regarding the user interfaces in an ES and a KBS, some of the following important points need to be considered [22].

- **Display type:** Early systems used text-based shells, which used simple command line character strings for input and output, whereas today, most ESs are GUI-based and offer bonuses in the various tools, like textboxes, graphics, dialog boxes, dropdown menus, etc.
- **Information entry:** Information can be entered into the system in different ways. The system may communicate with users either subjectively or objectively. It may ask a question or provide a form to be filled in. The question may require a single answer or multiple answers.
- **Information display:** The ES can display the information through textboxes, message boxes, graphics, radio buttons, etc. The ES must be able to give the final conclusion or findings to the user.
- **Interface Control:** The developed systems must be able to control the session, including its start, termination, suspension and resumption. The system must provide the configuration capabilities to the user through radio buttons, checkboxes and buttons.

Some authors categorized shells into two types: (a) rule-based network shells and (b) pattern-matching shells [20]. Rule-based networks limit the rules in a decision tree on the way to a single conclusion by using the backward chaining method of inference. Examples of these shells are CRYSTAL and VP-EXPERT. Pattern-matching shells are comparatively permissive. Rules contain variables and can be set for either backward or forward chaining. These shells use a typical pattern-matching mechanism to provide a speedy inference process. Well-known examples of pattern-matching shells are Xi Plus and Savoir.

■ Self-Learning

The system's self-learning is a scientific practice that relies on the algorithms explicitly used for it. The system evolves according to the experts' and the end users' input and processed results. Sometimes, it becomes impossible for the developers to embed all the possibilities inside the system; rather, they develop a mechanism to make the system able to upgrade the knowledge by itself. A research system [23] uses a learning engine that learns new rules from the shell of any domain. The new rules can be modified, updated or deleted by the experts for validation of rules or facts.

The principal gain from self-learning is that the system does not prompt the user for input and executes highly frequent primary rules in a specific domain. However, if the system has some highly critical nodes, then the developer must be very careful to permanently enable these nodes for execution [24].

Development

KBSs can be built for countless fields of life, such as medicine, astronomy, production plants, business, recruitment, the web, scheduling, etc. The history of these systems begins with the foundation of AI. Lots of research has been done in the field of KBSs, like other areas of the computer sciences. Some of the published research work is described in the following.

- Mishra [25] proposes an effective solution for wireless network security threats like worms, viruses, sniffing, network eavesdropping, etc. The traditional mechanism does not offer evaluation of threats, and thus, security of the entire system cannot be achieved. A network security context is created by mining data about the occurrence of attack patterns, based on a knowledge discovery approach. According to the author, intrusion detection systems (IDSs) have preconfigured rules through which they generate true (and plenty of false) alarms. The proposed system defines situation awareness (SA) as the perception of elements in the environment. There are four levels in SA: perception, comprehension, projection and resolution. A security pattern-mining algorithm is used for extraction of knowledge about security from the alert events.
- According to Ahmed et al. [26], the data gathered from the internet, in response to any query or search, is vague or irrelevant most of the time. Therefore, there is a need for technology to be developed that offers the extraction of precise and focused information. In order to solve this problem, a knowledge-based grid in a semantic web, combined with the functionalities of a traditional search mechanism, can be used. The paper focuses on serving university students so they can gather knowledge related to their current academic studies and to help them in their future education. The four major components of this system are: (a) a semantic web interface, (b) an ontology server, (c) a knowledge server, and (d) a knowledge directory server.
- Kerzazi et al. [27] presented an addition to the Software and System Process Engineering Metamodel (SPEM) with a knowledgebase for explicit representation of knowledge within the process model. This system handles knowledge dependencies in the software process model by providing data related to the lack of knowledge about

any activity present in the model. An ontological domain, using a tree of interrelated concepts, is developed for building knowledge repositories.

- The work by Agrawal and Siddiqui [28] is used to conclude the sentiment orientation of sentences by using a knowledge-based system with SentiWordNet and a linguistic heuristic for star rating reviews. Unlike earlier work, where a knowledgebase is used for positive or negative classifications, this paper presents a technique to strengthen the judgment on a five-point scale. The authors propose three different techniques for reviews' ratings generation. They are (a) the SentiWordNet Average Scoring Approach, (b) the parts-of-speech–based approach, and (c) a context-sensitive approach used for rating-generation purposes.
- Considerably important things related to deep drawing die are the selection of suitable types and sizes of the die components. There are a number of die components available, from which the designer needs to choose. Simply, we can say that selection of a die is not an easy job, and the designer needs expertise or experts' given knowledge. Naranje and Kumar [29] proposed a KBS that can be used in the selection process for major components of a deep drawing die. The system is structured in eight modules. In each module IF-THEN production rules are used to get knowledge about the die from experts.
- Researchers studied the following three different e-learning tools, which are referred to as learning management systems (LMSs): Blackboard, Moodle and Claroline. In addition, they seek to develop a more efficient and inexpensive system for providing the same usefulness in the collective services of the present e-learning tools. The existing systems were tested for their considerable features, like UI functionality, usability, reliability, portability and compatibility. The Integrated Dynamic Learning Environment (IDLE) system analyzes these functionalities in a larger context, along with a list of problems encountered during the response from the user through internet [30].
- Vecchia et al. [31] discuss major security concerns about radio frequency identification (RFID) systems, specifically, illegal cloning of RFID tags. Attackers can clone a tag and use it without the consent of the tag owner for unauthorized access to secured facilities, to disrupt the supply chain and for fraudulent purchases. To identify tag cloning, a knowledge-based system is proposed, which works on a track-and-trace mechanism. A semantic web language is used to formalize the track-and-trace model stored in the knowledgebase to attain a well-defined, unambiguous and machine-readable knowledge representation.
- Heeptaisong and Shivihok [32] worked on the development of a KBS for retrieval of knowledge regarding soil (useful for farmers) from different sources on the internet. According to the authors, the knowledge available has various formats, like HTML, databases and digital libraries on the internet. The research framework, using ontologies, consists of three major components: (a) feature extraction and knowledge input processing, (b) an automatic term weight process, and (c) a knowledge-retrieval processing

The knowledge stored in the human brain is in passive and abstracted form. There is no mechanism to acquire knowledge by simply scanning or inserting a connection to the expert's brain. In most cases, the experts cannot deliver all their knowledge in a specific domain, and it is also possible that they may not be familiar with the KBS and its development. It is the responsibility of the knowledge engineer to provide an interface and to interrogate the experts in such a way that most of their problem domain–related knowledge is gathered.

Knowledge from experts to the users through the KBS goes through the following three stages.

- Knowledge acquisition: includes gathering knowledge in the form of facts through different methods, like observations, interviews, simple close-ended questionnaires, etc.
- Knowledge verification: is when knowledge engineers or developers apply different schemes, like rules, frames, and semantics, to make the knowledge friendly for the development process.
- Knowledge representation: is the presentable format of the knowledge—understandable and in accordance with the problem domain for the user.

In 1989, a study at the University of Amsterdam was carried out on inefficiency in methods, which caused insignificant development of expert systems, particularly the knowledgebases. As a result, the structured method and a principled approach for developing a KBS was developed, named Knowledge Acquisition and Documentation Structuring (KADS). The KADS approach comprises three major knowledge acquisition tasks: elicitation, analysis and formalization. Two key features of KADS are the use of multiple models to ease the knowledge engineering process and the description as immediate models for expertise data and system design.

We can categorize the development of KBSs under development tools, languages and shells. A KBS tool is a computer software program or utility package to obtain, process, store and reuse the experts' knowledge to develop a KBS with the assistance of prewritten code. Examples of these tools are LOOPS, Art and KEE. Well-known popular programming languages, like C++, Java and Microsoft's .Net framework, can also be considered for creating a new KBS. However, the built-in support of these languages for intelligent systems is limited, and the developer needs to code the intelligence from scratch. AI logic programming languages are used for building KBSs. Symbolic logic programming languages provide rule-based procedures or functions for deducing new facts from existing input data. These languages offer special support for building a knowledgebase that provides functionalities for accessing, deducing and storing knowledge facts. Examples of logic programming languages are Prolog and Lisp. Usually, ES shells are ready-to-use ESs with a blank knowledgebase. These shells provide a real-time rule-based inference processing mechanism and knowledge representation facilities. Examples of such shells are Crystal, XpertRule, Leonardo and Xi-Plus.

Some examples of KBS development languages, shells and tools are elaborated in the following.

- *C Language* Integrated Production System (CLIPS): This is a public domain rule-based programming language and a widely used ES development tool first created in 1985 by NASA under a project named NASA's AI Language (NAIL). Since it is written in C, it can be installed on any system having an ANSI-compliant C or C++ compiler. CLIPS uses a simple editor for all operations. It implements a pattern-matching Rete algorithm for inference. CLIPS works on three main components: facts, rules and agenda.
- Java Expert Systems Shell (JESS): First developed in 1995, this rule engine is a modified flavor of the CLIPS programming language, with some additional features and capabilities. JESS was developed in Java by Ernest Friendman-Hill at Sandia National Laboratories. JESS is a tiny and fast language with power to access all Java APIs. It uses an advanced version of the Rete algorithm. It supports many operating systems, including Android. JESS can be used in command-line and GUI applications, applets and servlets. JESS owns JessML, a declarative XML rule-based language. The fundamental units of code in JESS are list-enclosed sets of parentheses.
- Prolog: Developed by Colomerauer and Philippe Roussel in 1972, Prolog is a much admired logic programming language, which can be used for making KBSs. It provides an easy way to represent knowledge in AI applications. Typically, Prolog provides a text-based interface, but modern Prolog gives a complete GUI development environment. It is based on Horn clause logic. In this language, data is formed in constants, variables and structures called terms. It represents rules, facts and express relationships. Queries can be applied on facts to sort out a result, either true or false. Prolog is a declarative, object-oriented, imperative and functional logic programming language.
- Lisp: The list processor, first used by MIT in 1958, is an old, but still-practised general-purpose language. Nowadays, Lisp is used to develop AI applications, like KBSs, ESs, natural language processing (NLP), etc. Being an early language, it introduced many ideas, such as dynamic typing, the tree structure, automatic storage management, and many more, into the computer sciences. Lisp has many variants, which are commercially used. Some of these known dialects are Common Lisp, Clojure, Emacs Lisp and Scheme. It has a parenthesized polish notation. The main data structure in Lisp is linked lists. In Lisp, the source code is considered a data structure, and one can effortlessly define executable data structures in it.
- FLEX: This is an ES development toolkit owned by Logic Programming Associates (LPA). It was developed in Prolog and uses English-like natural language syntax, easily understandable by experts. FLEX delivers support for both types of inferences (forward chaining and backward chaining). This tool offers multicolored rich syntax editing in the development environment, which uses different colors for the identification of classes, numbers, data types, etc., in the code. It has a range of features, like rule-based programming, inheritance and data-driven procedures. FLEX uses a C++ class-like data structure known as frames.
- Gensym G2: This tool is considered one of the powerful real-time ESs. It helps to develop, install and familiarize real-time, rule-driven intelligent systems. It can be used for KBS decision-making applications for finance, military, utilities and telecommunication etc. G2 can be connected to different databases, programs, applications and systems by using various adaptors and interfaces. It can be used on most versions of Microsoft Windows and Red Hat Enterprise Linux 5.6.
- XpertRule: This was developed by a UK-based company named Attar Software for building KBSs and business rule management (BRM). This tool has a companion program known as XpertMiner, which spots patterns in stored data and helps users create data-mining processes. This tool is Microsoft Windows–based and provides a graphic interface.
- GoldWorks III: Owned by GoldHill Inc., this is one of the pioneer Windows-based development tools for expert systems, which can be used for construction of a KBS. It provides a GUI for rapid and easy development. It is based on the OOP paradigm, Common Lisp Object System (CLOS). GoldWorks III can access non-native programs, databases and applications by using Microsoft's Dynamic Data Exchange (DDE) interface. It provides a hybrid of rule-based and frame-based methods of knowledge representation.
- Vidwan: This is an ES shell, can be used for KBS development, and was created in 1993 at the National Centre of Software Technology (NCST), India. It works on backward chaining, builds knowledge in an IF-THEN structure and offers a reasoning facility through questions and explanations. Vidwan provides an interactive GUI for development under Microsoft Windows and Unix. It works on the rule-based components of a domain's inner

parameters of interest and the relationships among these parameters. The parameters used in a medical diagnosis system (e.g. the age of a person and symptoms reported) are identified as attributes in Vidwan. For formalizing relationships among these attributes, the IF-THEN structure is used.

Issues

In addition to numerous advantages to using a KBS, there exist some serious issues regarding the successful development, deployment and maintenance of these systems. Issues in knowledge acquisition, knowledge verification, knowledge representation, deployment and maintenance are discussed in the subsequent sections.

Knowledge Acquisition

There are two stakeholders for a specific problem domain in this phase: the knowledge engineer and the experts. The knowledge acquisition process is completely influenced by the platform provided by the knowledge engineer [33]. Manual and interactive computer-based techniques can be used for contacting the experts and digging out their expertise in a specific domain. There are many issues regarding knowledge acquisition from the experts. Some of these issues are as follows [34].

- Most of the knowledge inhabits the expert brain in passive form. There is no mechanism to drain all domain-related knowledge with the assistance of questions or interviews.
- The questionnaire for getting knowledge is incomplete most of the time, because it is impossible for the knowledge engineer to think of all the possibilities. Moreover, experts keep a vast amount of knowledge with them and cannot provide all their domain-related knowledge without queries.
- The communication gap between the knowledge engineer and the expert is also a considerable issue. Differences in language and misconceptions can influence the knowledge acquisition process. Also, due to the comparatively minor relevance of knowledge engineers, they may not understand the basic terms used in a domain.
- Sometimes the experts are not accustomed to computer-based systems, and thus, they may be reluctant to provide domain knowledge. Moreover, there is a possibility that the experts will not cooperate, believing that the system can undermine, or misuse, their expertise.
- The time factor can also affect the knowledge acquisition process. Experts having a busy life sometimes may not be able to respond in the time frame set by the knowledge engineer.
- Tacit knowledge cannot be described or explained. Experts may practise something, but do not know how to transfer or describe it to others.
- The system engineer might hire an expert who does not exactly know (or lacks some knowledge about) the domain. It is also very difficult for the knowledge engineer to know the capabilities of the expert in advance.

Knowledge Verification

The knowledge engineer or developer verifies and validates the input or gathered knowledge for further discourse. Varieties of issues exist in this process. Some problems were described by Paul et al. [35] as follows.

- In most cases, the knowledge engineer finds the knowledge in raw, vague or abstracted form, and he or she needs some extra assistance from the experts to understand it.
- The criteria for determining the correctness of knowledge may change as the system evolves. Each problem consists of further sub-problems. Any change in the state of a sub-problem directly affects the main criteria for verification of the knowledge, as required for the whole system.
- Sometimes two or more rules overlap. Passing them all may cause redundant storage in the knowledgebase.

Knowledge Representation

Knowledge representation is a procedure for operating on the knowledge. It can be presented in various ways, depending upon its type. Rules, frames, logic and semantic networks are used for knowledge presentation. Some of the known issues in this area are explained in the following.

- The system may not understand, or may not be able to process, the gathered high-level abstracted knowledge. Additional effort is required to convert such knowledge into primitive form. Another issue arises with this action, i.e. the basic form of knowledge requires massive storage.
- One issue in knowledge representation is that the system uses a default reasoning mechanism for tackling problems. The representation may be wrong if there are some exceptions regarding that knowledge. In the real world, we can face violations of different rules in various situations due to different exceptions [36][37].
- A common person knows countless atomic facts. It is impractical for a developer to represent a huge number of facts in the system. It is not possible to input each and every fact in the knowledgebase.
- Identification of relevant knowledge for representation is also a challenge for the system. In most cases, the system will not be able to recognize the depth of knowledge and differentiate it from other knowledge [38].

■ Deployment

Some major issues faced in the deployment stage of the KBS life cycle are as follows [39].

- A huge knowledgebase with stored data consumes extensive computer resources. It also distresses the overall speed of the system. Such a KBS requires expensive and more efficient hardware, particularly if it performs in a real-time environment.
- Most customers are not familiar with the languages and tools used for building a KBS, like Lisp, JESS, CLIPS, etc. This is also a big problem when operating such systems.
- There is no CASE methodology to create standards for developing a KBS. Methods for testing a KBS to find out the extent of its applicability are also not well recognized.

■ Maintenance

For maintenance purposes, the users of the system are trained, and proper documentation is given to the organization. Still, the developer can face some issues. Issues related to maintenance are as follows [40] [41].

- KBSs are built to work in a dynamic manner, in order for the system to remain relevant. It must adjust to changing conditions by itself, in order to achieve the same goals. These systems cannot adapt by themselves in most cases. For the developer, it is also very difficult to recheck and alter the complex structure of a developed system.
- The maintenance of the knowledgebase is completely dependent on the experts. They could retire, leave the job or die, causing a critical problem for maintenance.
- Most of the time, the experts abandon the knowledge engineers, leaving them all responsibility for maintenance of the deployed system.

Conclusion and Future Work

In this paper, we discussed KBSs in detail, along with examples of a few proposed systems, tools, languages and shells used for development, and we discussed issues regarding the development, deployment and maintenance of these systems. KBSs have made a significant contribution in almost all computer-aided fields of life. These systems are very useful, and help users in unhandled and abnormal situations. Integration of knowledgebases with other systems provides significant advantages in overall quality of computing and management. KBS systems can be deployed everywhere, in everything, and be utilized anytime. These systems are used in smart phones for a variety of purposes. Development of lightweight, efficient and real-time applications in handheld devices has vast potential. These KBSs may utilize user behavior, different sensors and functionalities, and must consider the limitations of smart phones. Networked KBSs in many domain, are in use and can evolve and be expanded in accordance with innovations. KBSs are used by many software development tools, especially in the design process of the software development life cycle. These systems can be made more proficient and specific to new research. Developers can use knowledgebases for redesigning the performance of already-built software via reverse software engineering. Expert approaches can be shared among different software developers in different ways. Many KBSs are applied in the field of medical sciences. These systems can be further improved and provide better clinical diagnoses. Knowledgebases can be used in data mining and data warehousing for various purposes and outcomes. These applications can be made more intelligent and efficient as well.

References

[1] U. Marvis, K.Shade, F. Atisi and B. Olayiwola. “Overview of knowledge-based system,” *International Journal of Computer & Communication Engineering Research* vol. 2, no.3, pp.125-129, 2014. Article (CrossRef Link)

[2] U. Marvis, K.Shade, F. Atisi and B. Olayiwola. “Overview of knowledge-based system,” *International Journal of Computer & Communication Engineering Research* vol. 2, no.3, pp.125-129, 2014. Article (CrossRef Link)

[3] A. Assawamakin, N. Chalortham, T. Ruangrajitpakorn, C. Limwongse, T. Supnithi, S. Tongsima, “A development of knowledge representation for thalassemia prevention and control program,” *2011 7th International Conference on Natural Language Processing and Knowledge Engineering*, Nov. 2011. Article (CrossRef Link)

[4] W. J. Clancey., "12 Acquiring, representing, and evaluating a competence model of diagnostic strategy," *The nature of expertise*, pp. 343, 2014. Article (CrossRef Link)

[5] P. O. Wennerberg, "Ontology based knowledge discovery in social networks," *JRC Joint Research Center, European Commission, Institute for the Protection and Security of the Citizen (IPSC)*, Tech. Rep., 2005. Article (CrossRef Link)

[6] J. W. Eggstaff, T. A. Mazzuchi, S. Sarkani, “The effect of the number of seed variables on the performance of Cooke′s classical model,” *Reliability Engineering & System Safety*, vol. 121, pp. 72–82, Jan. 2014. Article (CrossRef Link)

[7] M. Henrion, “Propagating uncertainty in Bayesian networks by probabilistic logic sampling,” *Machine Intelligence and Pattern Recognition*, pp. 149-163, 1988. Article (CrossRef Link)

[8] A. Baltag, B. Renne, S. Smets, “The logic of justified belief, explicit knowledge, and conclusive evidence,” *Annals of Pure and Applied Logic*, vol. 165, no. 1, pp. 49-81, Jan. 2014. Article (CrossRef Link)

[9] Y. Gong, M. Janssen, “An interoperable architecture and principles for implementing strategy and policy in operational processes,” *Computers in Industry*, vol. 64, no. 8, pp. 912-924, Oct. 2013. Article (CrossRef Link)

[10] A. Mehmood, S. Khan, F. Khan, M. Yaseen, M. Majeed, “Inter-operability for open clouds VIA adaptive transport protocols at multi-agent systems (MASS),” *International Conference on Computer Technology and Development, 3rd (ICCTD 2011)*, pp. 2303-2307, Jan. 2011. Article (CrossRef Link)

[11] G. Halit Soydan, M. M. Kokar, “A partial formalization of the CMMI-DEV—A capability maturity model for development,” *Journal of Software Engineering and Applications*, vol. 05, no. 10, pp. 777-788, 2012. Article (CrossRef Link)

[12] J. Urbani, S. Kotoulas, J. Maassen, F. Van Harmelen, H. Bal, “WebPIE: A web-scale parallel inference engine using MapReduce,” *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 10, pp. 59-75, Jan. 2012. Article (CrossRef Link)

[13] M. D. Hoffman, D. M. Blei, C. Wang, J. Paisley, "Stochastic variational inference," *The Journal of Machine Learning Research*, vol.14, no.1, pp. 1303-1347, 2013. Article (CrossRef Link)

[14] E. Balai, M. Gelfond, Y. Zhang, “Towards answer set programming with sorts,” *Lecture Notes in Computer Science*, pp. 135-147, 2013. Article (CrossRef Link)

[15] R. Datta and J. Z. Wang, "ACQUINE: aesthetic quality inference engine-real-time automatic rating of photo aesthetics," *Proceedings of the international conference on Multimedia information retrieval*, (MIR '10), ACM, pp. 421-424, 2010. Article (CrossRef Link)

[16] A. Mehmood, H. Song, J. Lloret, “Multi-agent based framework for secure and reliable communication among open clouds,” *Network Protocols and Algorithms*, vol. 6, no. 4, p. 60, Dec. 2014. Article (CrossRef Link)

[17] S. Roy and M. F. Yan, "Method and system for creating OWL ontology from java," *U.S. Patent No. 8*,656,356. 18 Feb. 2014. Article (CrossRef Link)

[18] D. Gaševic, D. Djuric, V. Devedžic, “Ontologies,” *Model Driven Engineering and Ontology Development*, pp. 45-80, 2009. Article (CrossRef Link)

[19] I. Horrocks, U. Sattler, “A tableau decision procedure for SHOIQ,” *Journal of Automated Reasoning*, vol. 39, no. 3, pp. 249-276, Jul. 2007. Article (CrossRef Link)

[20] T. J. M Bench-Capon. *Knowledge-based systems and legal applications*. vol. 36. Academic Press, 2014. Article (CrossRef Link)

[21] S. Liu, P. Zaraté, “Knowledge based decision support systems: A survey on technologies and application domains,” *Lecture Notes in Business Information Processing*, pp. 62–72, 2014. Article (CrossRef Link)

[22] A. Ligęza, G. J. Nalepa, “A study of methodological issues in design and development of rule-based systems: proposal of a new approach,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 2, pp. 117-137, Feb. 2011. Article (CrossRef Link)

[23] A. Mehmood, A. Ghafoor, H. F. Ahmed, Z. Iqbal, “Adaptive transport protocols in multi agent system,” *Fifth*

International Conference on Information Technology: New Generations (itng 2008), Apr. 2008. Article (CrossRef Link)

[24] A. Mehmood, S. Khan, B. Shams, J. Lloret, "Energy-efficient multi-level and distance-aware clustering mechanism for WSNs," *International Journal of Communication Systems*, vol. 28, no. 5, pp. 972–989, Dec. 2013. Article (CrossRef Link)

[25] D. U. Mishra, "Knowledge based context awareness network security for wireless networks," *International Journal of Computer Trends and Technology*, vol. 4, no. 10, pp. 3751-3757, 2013. Article (CrossRef Link)

[26] M. Ahmed, S. Zeeshan Hussain, S. A. M. Rizvi, "Knowledge based grid using semantic web for the university system," *International Journal of Computer Applications*, vol. 76, no. 15, pp. 46–52, Aug. 2013. Article (CrossRef Link)

[27] N. Kerzazi, M. Lavallée, P. N. Robillard, "A knowledge-based perspective for software process modeling," *e-Informatica Software Engineering Journal*, vol. 7, no.1, 2013. Article (CrossRef Link)

[28] S. Agrawal, T. J. Siddiqui., "Feature based star rating of reviews: A knowledge-based approach for document sentiment classification," *International Journal of Hybrid Information Technology*, vol. 5.no. 4, pp. 95-110, 2012. Article (CrossRef Link)

[29] V. Naranje, S. Kumar, "A knowledge based system for selection of components of deep drawing die," *American Journal of Intelligent Systems*, vol. 2, no. 2, pp. 1–11, Mar. 2012. Article (CrossRef Link)

[30] M. V. Abante, L. W. Rabago, B. T. Tanguilig. "Mapping of the three different E-Learning management tools: A basis for integrated development learning environment (IDLE)," *The Third International Conference on E-Learning and E-Technologies in Education (ICEEE2014)*. The Society of Digital Information and Wireless Communication, pp. 182-190, 2014. Article (CrossRef Link)

[31] G. Della and M. Esposito, "A knowledge-based approach for detecting misuses in RFID systems," *Designing and Deploying RFID Applications*, Jun. 2011. Article (CrossRef Link)

[32] T. Heeptaisongand A. Shivihok., "Soil knowledge-based systems using ontology," *Proceedings of the International MultiConference of Engineers and Computer Scientists*. vol. 1. 2012. Article (CrossRef Link)

[33] M. Galetakis, S. Papadopoulos, A. Vasiliou, C. P. Roumpos, T. Michalakopoulos, "Development of an expert system for the prediction of the performance of bucket-wheel excavators used for the selective mining of multiple-layered lignite deposits," *Proceedings of the 12th International Symposium Continuous Surface Mining - Aachen 2014*, pp. 47–58, 2015. Article (CrossRef Link)

[34] Y. Erdani, "Developing backward chaining algorithm of inference engine in ternary grid expert system," *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 9, 2012. Article (CrossRef Link)

[35] S. Paul, et al. "A semantically enabled formalism for the knowledge management of Parkinson's disease," *Informatics for Health and Social Care*, vol. 31, no.2, pp. 101-120, 2006. Article (CrossRef Link)

[36] P. Erola, S. Gómez, and A. Arenas, "On the routability of the internet," *Modeling and Simulation in Science, Engineering and Technology*, pp. 41-54, 2013. Article (CrossRef Link)

[37] A. Mehmood, H. Song, "Smart Energy Efficient Hierarchical Data Gathering Protocols for Wireless Sensor Networks," SmartCR, vol. 5, no. 5, pp. 425-462, Oct. 2015. Article (CrossRef Link)

[38] V. V. M. Rao, V. V. Kumari. "An analysis and knowledge representation system to attain the genuine web user usage behavior," *International Journal on Computer Science & Engineering*, vol. 5, no. 5, pp. 302-316, 2013. Article (CrossRef Link)

[39] G. C. Kane, M. Alavi, G. Labianca, S. Borgatti, "What's different about social media networks? A framework and research agenda," *MIS Quarterly, forthcoming*, vol. 38, no. 1, pp. 275-304, 2012. Article (CrossRef Link)

[40] M. Eaton, "Evolutionary humanoid robotics (EHR)," Springer Briefs in Intelligent Systems, pp. 59-83, 2015. Article (CrossRef Link)

[41] W. He, "Improving user experience with case-based reasoning systems using text mining and Web 2.0," *Expert Systems with Applications*, vol. 40, no. 2, pp. 500-507, Feb. 2013. Article (CrossRef Link)

Muhammad Muneer Umar is currently doing his PhD at Institute of Information Technology, Kohat University of Science and Technology, Pakistan. He taught KBS as a subject to undergraduate students of Information Systems at King Khalid University, Saudi Arabia for five academic years. Presently, he is doing research in routing protocols for Mobile Ad-hoc Networks and Wireless Sensor Networks.

Amjad Mehmood received the Ph.D. degree in Wireless Networks from Kohat University of Science & Technology, Kohat, in 2014, and the MS degree in Distributed Computing from Kohat University of Science & Technology, Kohat. In 2003, he joined Kohat University of Science & Technology, Kohat, KP, where he's been currently serving as a senior faculty member and the Coordinator of MS/PhD program. Dr. Amjad is interested to work in the areas of cyber-physical systems, internet of things, connected vehicles, wireless communications and networking, optical communications and networking, smart grid communications and networking, and body sensor networks. His research wok is supported by the Institute of information Technology at KUST. Dr. Amjad's supervised many students of BCS, MCS, MS and PhD in the above mentioned interests. He's also been remained the part of organizing different workshops, seminar and training sessions on different technologies at the University under the umbrella of Computer Science Society of IIT, KUST. Dr. Amajd has served on the technical program committee for numerous international conferences, including CCNC, SCPA, INFOCOM, SCAN, and so on. Dr. Amjad is a reviewer or associate editor for many peer-reviewed international journals. Dr. Amjad has published more than 30 academic articles in peer-reviewed international journals and conferences around the world.

Houbing Song. (M'12-SM'14) received the Ph.D. degree in electrical engineering from the University of Virginia, Charlottesville, VA, in August 2012, and the MS degree in civil engineering from the University of Texas, El Paso, TX, in December 2006. In August 2012, he joined the Department of Electrical and Computer Engineering, West Virginia University, Montgomery, WV, where he is currently an Assistant Professor and the founding director of both the West Virginia Center of Excellence for Cyber-Physical Systems (WVCECPS) sponsored by the West Virginia Higher Education Policy Commission, and the Security and Optimization for Networked Globe Laboratory (SONG Lab). His research interests lie in the areas of cyber-physical systems, internet of things, cloud computing, big data, connected vehicle, wireless communications and networking, and optical communications and networking. Dr. Song's research has been supported by the West Virginia Higher Education Policy Commission, Virginia Department of Transportation, and CONSOL Energy Inc. Dr. Song is a senior member of IEEE and a member of ACM. Dr. Song is an associate editor for several international journals, including IEEE Access and KSII Transactions on Internet and Information Systems, and a guest editor of several special issues. Dr. Song was the general chair of 5 international workshops, including the first IEEE International Workshop on Security and Privacy for Internet of Things and Cyber-Physical Systems (IOT/CPS-Security 2015), held in London, UK, 2nd International Workshop on Mobile Cloud Computing systems, Management, and Security (MCSMS-2016), to be held in Madrid, Spain, and the first/second/third IEEE ICCC International Workshop on Internet of Things (IOT 2013/2014/2015), held in Xi'an/Shanghai/Shenzhen, China. Dr. Song also served as the technical program committee chair of the fourth IEEE International Workshop on Cloud Computing Systems, Networks, and Applications (CCSNA), held in San Diego, USA. Dr. Song has served on the technical program committee for numerous international conferences, including ICC, GLOBECOM, INFOCOM, WCNC, and so on. Dr. Song has published more than 80 academic papers in peer-reviewed international journals and conferences. One of his papers was selected as the focus paper highlighted in IEEE ComSoc Technology News (CTN).