Basic Concepts of Expert System Shells and an Efficient Model for Knowledge Acquisition

Emmanuel C. Ogu[1], Adekunle, Y.A.[2]

[1, 2]Department of Computer Science,
Babcock University, Ilishan-Remo, Ogun State, Nigeria

[1]*ecoxd1@hotmail.com*, [2]*adekunleya@gmail.com*

Abstract: ***An Expert System is an intelligent computer program that employs knowledge and inference procedures to solve problems that are considered difficult enough to require significant human expertise for their solutions*** *[12]****. Since the implementation of the first expert systems (the DENDRAL, MYCIN, Dipmeter Advisor, XCON, to mention but a few.) in the early 1970s, and the successes that followed, coupled with the advantages that humankind benefited from these early expert systems and their successors; one would expect that four decades later, there would have been a boom in the use of expert systems to perform more specialized tasks in different specific domains and application areas. The reasons for this could stem, asides other non-intrinsically-technical factors (such as financial implication, cultural and religious beliefs and restrictions surrounding the implementation of Expert Systems especially in third-world African Countries), either from a lack of adequate knowledge of the existence of Expert System Shells and/or the facilities, utilities and tools they provide that make the development of new expert systems sort of a "pretty easy endeavour". This paper lays emphasis on the basic concept of Expert System Shells with the aim of highlighting their basic usage and creating some form of in-depth generic exposé into some of the component facilities and utilities they provide; proposing a more efficient model for Knowledge Acquisition – which would make the Knowledge Acquisition Bottleneck of developing Expert Systems a thing of no more concern – as well as highlight the functions performed by Expert System Shells, in a bid to emphasize their reusability across different other specific application areas either within or outside the same domain.***

Keywords: Knowledge-Based Expert Systems, Expert Domain Areas, Domain Application Areas, Symbolic Information and Reasoning, Knowledge Acquisition.

1. Introduction

[1], [2] What is an Expert System?
Many definitions exist for Expert Systems, some of the most popular of these being:

- An expert system is computer program that contains a significant portion of the specialized knowledge of a human expert in a specific, narrow domain, and emulates the decision-making ability of the human expert in that area [22].
- An expert system is a program intended to make reasoned judgements or give assistance in a complex area in which human skills are fallible or scarce [6].
- An expert system is a program designed to solve problems at a level comparable to that of a human expert in a given domain [3].
- An expert system is a computer program (or system) that operates by applying an inference mechanism to a body of specialist expertise represented in the form of "knowledge" [4].

One of the most widely acceptable definitions of expert systems, however, was given by Edward Feigenbaum of Stanford University. He defined an Expert System as "an intelligent computer program that employs knowledge and inference procedures to solve problems that are considered difficult enough to require significant human expertise for their solutions [12]."

Before proceeding, the following terms must also be understood:

- Language: is basically the translator of a command written in a specific syntax. The Expert System Language provides an Inference Engine to execute the statements of the Language.
- Tool: this refers to the Language plus associated utility software for debugging, developing, and deploying computer applications. Examples: text editors, (cross) compilers, code generators (which can generate code from tables e.g. XTRAN and similar others). Some tools may also be integrated with all its utility programs into a single integrated user interface.

Typically, an Expert System is composed of two major parts: the Knowledge-base, and the Expert System Shell. The diagram below shows the typical Expert System Architecture:

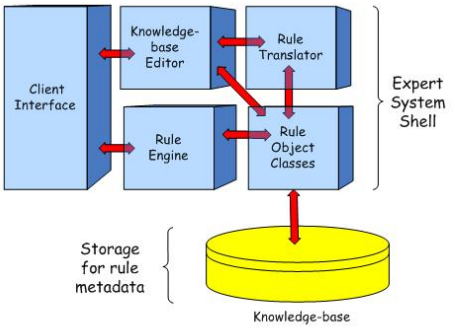

Figure 1: Generic Expert System Architecture [10].

2. Current Model for Knowledge Acquisition

The process of building an expert system is commonly referred to as *knowledge engineering*. This essentially implies knowledge acquisition from a human expert or other source(s) and then coding / representing such knowledge in

the knowledge base of the expert system. The main phases in the process of Knowledge Acquisition are [9]:

- The dialog process which involves the knowledge engineer acquiring knowledge from the human expert in the domain and explicitly coding it into the expert system knowledge base (the process known as *Knowledge Engineering*).
- After the coding, the human expert evaluates the expert system and gives feedback or criticisms to the knowledge engineer.
- The knowledge engineer then modifies the knowledge base to reflect the human expert's comments

The knowledge engineering process is illustrated in the diagram below:

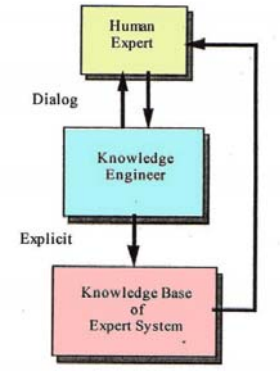

Figure 2: The Knowledge Engineering Process [9]

3. Review of Related Researches

Sener (1991) observed that though humans long(ed) to be able to have the experts they desire at their disposal through the use of Expert Systems, expert system applications have mostly remained in the domain of those who are highly skilled in programming [14]. In 1992, Susan M. Land conducted a research on Expert System Design Shells with a primary focus on its role in solving problems in the field of Education and Learning [16]. In 1995, L.K. Woolery, D.S. Harr & D.L. Harr developed an object oriented expert system shell called "The Gestation Predictor" for use on pregnant women by clinical professionals during their preterm birth screening [5]. In 2002, MD Salim, A. Villavicencio, & M. A. Timmerman described a variant of the Function Point Analysis Methodology (developed by Albrecht in 1979) – for assessing a software package that can measure both the accuracy and user friendliness of an Expert System Shell – for the evaluation of Expert System Shells in Industrial Technology Pedagogy [7]. In 2012, M.N.K. Boulos demonstrated the feasibility of using expert system shells for rapid clinical decision support module development in which an expert system shell was used to build a simple rule-based system for the crude diagnosis of vaginal discharge [8]. In January 2013, Samawi, et al observed that many expert system programmers suffer from knowledge acquisition complications; adding that Expert System shells contain facilities that can simplify knowledge acquisition in such a way that domain experts (human experts) themselves are responsible for knowledge engineering [19]. They went further to develop an Arabic Expert System Shell (AESS) which was capable of diagnosing diseases based on Natural Language. The system comprises two phases. In the first phase, automatic acquisition of the human expert knowledge in Natural Arabic is done and the knowledge acquired is analyzed using an Arabic morphological system which analyzes the given Arabic phrase and finds the required keywords (roots). The system contains the built-in required domain dictionary to be used by the Arabic morphological system. In the second phase, the major concern is with the design of the inference engine along with the user interface (based also on natural language) which uses a backward chaining method (end-user interface).

The Knowledge Acquisition Subsystem is of the AESS is given below:

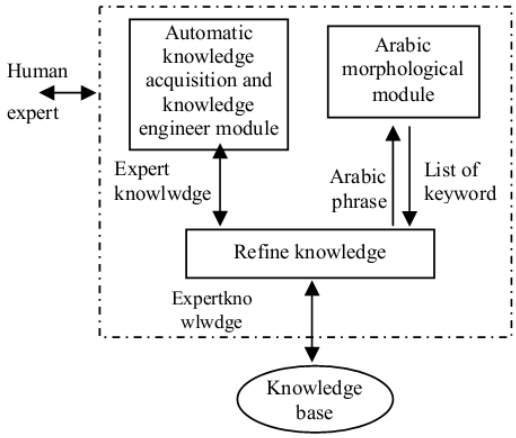

Figure 3: The Knowledge Acquisition Subsystem of the AESS [19]

When the system was tested by experts and end users alike, it was found to have a very exact performance in Knowledge-Base (KB) construction and problem diagnosis (with a diagnostic ability of 99%). They concluded hence, that the merging of a morphological system with knowledge acquisition is a very effective method for constructing target KBs without duplicate or inconsistent rules; further stating that the same technique could be employed in building expert system shells based on other natural languages – (such as English, French, Spanish, amongst others) – with the only difference being to build a suitable morphological system for that language along with the required domain dictionary [19]; a fact in line with which I align the context of this research – 100%. All these are but a "tiny nibble in the cake" when compared to the very extensive research and articles that have been undertaken and published (respectively) regarding the need for Expert System Shells in various specific domains, some of which resulted in the creation of these domain specialized ES Shells.

4. The Concept of an Expert System Shell

The Expert System Shell is essentially a special purpose tool that is built in line with the requirements and standards of particular domain or expert-knowledge area applications. It may be defined as a software package that facilitates the building of knowledge-based expert systems by providing a knowledge representation scheme and an inference engine [13]. The Shell refers to the software module containing an interface, an inference engine, and a structured skeleton of a knowledge base (in its empty state) with the appropriate knowledge representation facilities. In layman terms, it can be viewed as an empty bowl that is yet to be filled with the

expert knowledge elements that along with the inference engine may be used to process user requests to generate solutions to user problems [20]; In essence, any Computer Program which when supplied with a particular knowledge base yields an expert system is termed an Expert System Shell [11].

As was observed by Trollip & Lippert, (1987), Expert system shells provide methods of building expert systems without extensive knowledge of programming through mechanisms that:

1) Input the decisions, questions and rules that are followed;
2) Structure / Develop a knowledge database that can be manipulated by subsequent parts of the system;
3) Verify possible violations of surface validity; and
4) Operate the "inference engine" that operates on the rules, poses the questions to the users, and determines whether a particular decision is valid [18].

The Expert System Shell is responsible for managing and processing input and service requests from users and generating output; supporting the creation and modification of inference rules by knowledge engineers; translating the inference rules created into machine-readable forms; processing the information given by the user and the application layer modules, and relating such information to the concepts contained in the knowledge base through the inference rules; providing solutions for particular problems within the scope of its integrated knowledge domain; providing facilities for uncertain reasoning; providing facilities for knowledge representation (the knowledge representation knowledge) and editing the content of the knowledge base (knowledge base editor); and providing low-level support to expert system components (such as retrieving metadata from and saving metadata to the knowledge base, building Abstract Syntax Trees during the translation of inference rules, and other similar functions.) [11]. In essence, the system shell is indifferent to the rules it executes. This distinction is very important, because it means that the expert system shell can be applied to many different problem domains with little or no change. It also means that editing (adding or modifying) the rules of an expert system can implement some changes in the way the program behaves, without necessarily affecting the underlying (controlling component) – the shell.

One of the ways of building an expert system asides building it from scratch is that it can be built using a piece of an already developed software system which is known as a "tool", "skeleton" or "shell". Using these software systems, Expert Systems can be built that contain useful methods for solving problems without any domain-specific knowledge.

5. Components of an Expert System Shell

Some generic components of an Expert System Shell are given below:

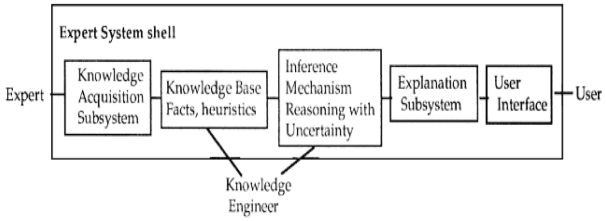

Figure 4: Generic Components of an Expert System Shell [22].

1. **Knowledge Base:** This is a store of factual and heuristic knowledge. An Expert System shell provides one or more knowledge representation schemes (in the form of knowledge bases) for expressing knowledge about the application domain. Some shells employ both Frames (objects) and IF-THEN rules in their knowledge base design. However, in PROLOG, the knowledge in the knowledge base is represented as logical statements.
2. **Reasoning or Inference Engine:** This refers to the Inference mechanisms that are used for manipulating the symbolic information and knowledge contained in the knowledge base in order to form a reasoning path towards the solution of a problem. The inference mechanism can range from simple *modus ponens* (a way that affirms by affirming) backward or forward chaining of IF-THEN rules to more complex case-based reasoning.
3. **Knowledge Acquisition Subsystem:** This is a subsystem that helps experts to build knowledge bases. It contains the Knowledge Representation Language. The process of collecting the relevant domain knowledge needed to solve problems and building (coding) the knowledge base (knowledge engineering) continues to present the biggest bottleneck in building expert systems.
4. **Explanation Subsystem:** This is a subsystem that explains the system's actions by trying to give reasons why a certain cause of action or a certain conclusion was chosen by an Expert System. The explanation can range from how the final or intermediate solutions were arrived at to justifying the need for additional data.
5. **User Interface:** This represents the means of communication with the user. The user interface is usually not a generic part of the Expert System technology, and was not given much attention in the early years of the development of Expert Systems. However, it is now widely accepted that the user interface can make a critical difference in the perceived utility of an Expert System regardless of the system's performance. The users enters a request in Natural Language, and the Expert System processes the Request to provide a result by using an Inference rule(s) to select the appropriate solution from a Knowledge Base.

Other Components may include:

6. **Trace Facility**: This is a component of an expert-system shell that allows following and inspecting the inference process [11].

The use of shells in designing Expert Systems carries with it many advantages. A system can be built to perform a unique task by programming the shell to contain all the necessary knowledge about a specific task domain. The inference engine which applies the knowledge to the task at hand

(may) already have been built into the shell. If the program is not too complex and the expert has had some training in the use of a shell, the expert can program the knowledge into the shell by himself, because every expert system shell or builder tool offers a formal language – a format for declarative knowledge in the knowledge base, known as the *knowledge-representation formalism*, for encoding the domain knowledge into the knowledge base [21].

6. An Efficient Model For Knowledge Acquisition

As was rightly observed by Sener (1991) that though humans long(ed) to be able to have the experts they desire at their disposal through the use of Expert Systems, expert system applications have mostly remained in the domain of those who are highly skilled in programming. The Knowledge Acquisition bottleneck has remained over the years because the gap between the Knowledge Engineer and the Human Expert has constantly increased. The Knowledge Engineer is perceived to always be some sort of expert programmer with a lot of specialized programming skills while the Human Expert has constantly been perceived (in most cases) to be an intrinsic IT-phobic, and as such, knowledge and inference rules could get into the knowledge base only through the Knowledge Engineer who in turn had to sit down for months (and sometimes years) to understand try to understand the skill(s) and technical language(s) of the Human Expert so that they could be adequately and accurately represented in the Knowledge Base using the correct set of inference rules and with minimal errors. This gap can only be closed when the Human Expert ceases to be an entire novice alien to the process and technique of Knowledge Engineering using Expert System Shells and the easy-to-use functionality and modules provided by these Shells. In addition, if Expert System Knowledge-representation formalisms could employ a formal language that tends to model very closely the Natural Language of man, such that most (or all) Expert System Shells would be able to provide a form of ultra-high level programming language abstraction – as a form of standardization that would be adhered to when creating these shells – which would conceal the explicit details of the underlying source codes and programming language (be it PROLOG, LISP or any of their variants), that power the shell, then humanity would experience the boom in the development, implementation and use of Expert Systems it has always longed for. I thus propose a model that emphasizes that with the above mentioned suggestions and standards in place, and with better understanding of the ease of programming expert systems using already developed shells in specific domain or application areas, the human experts would actually be able to carry out the process of Knowledge Engineering themselves, thus eliminating the Knowledge Acquisition Bottleneck in (a short) time to come, and resulting in a boom in the development and use of expert systems. This model is illustrated in the diagram below:

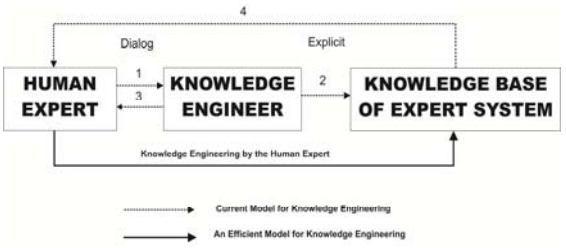

Figure 5: An Efficient Model for Knowledge Acquisition

7. Conclusion

Stylianou, Madey, and Smith (1992), noted that the selection of an adequate Expert System Shell is often the difference between a successful and an unsuccessful industrial application [15]. But then also, in two articles by Van Name and Catchings (1990), they observed that due to the large number of Expert System Shell packages on the market – and the lack of standardized means for describing the features and capabilities provided by these products – selecting an appropriate Shell package for an application can be "herculean task" even for experienced users.

There exist a lot of commercial and freeware shells today, generally ranging in size from shells on small PCs and workstations, to shells on larger mainframe computers; likewise ranging in complexity from the simple, forward-chained or backward-chained, rule-based systems requiring just a few days of training to those so complex that only highly trained knowledge engineers can use them to advantage; and in scope from general-purpose shells to custom shells tailored to perform a specific class of tasks, such as financial planning and auditing or process control in real-time.

Despite the fact that most Expert System Shells in use today help to simplify the task of programming expert systems (which is essentially the "heck" with producing Expert Systems), they generally do very little to help with knowledge acquisition and its associated bottleneck, which is a different concern entirely. *Knowledge acquisition* further refers to the task (or process) of furnishing expert systems with knowledge, a task that is currently performed by knowledge engineers alone. The choice of the reasoning method, or the shell, is quite important, but it may not be as important as the task of accumulating high-quality knowledge required in Expert Systems. The power of an expert system lies in its store of knowledge about the specific task domain because the more the knowledge an expert system has, the more competent it becomes.

Stylianou, Madey, and Smith (1992) further identified the following eight (8) categories of criteria for evaluating Expert System Shells: End User Interface, Developer Interface, System Interface, Inference Engine, Knowledge Base, Data Interface, Cost, and other Vendor-Related aspects [15].

In summary, generically, Expert System Shells can be reusable across many other application areas as soon as its knowledge base is emptied – as was the case with the MYCIN in which the knowledge base was emptied to form

the EMYCIN (Empty MYCIN) shell (the very first expert system shell – was created in 1970). This made the shell more useable across other application areas within the same domain. The implementation of the MYCIN Shell in other areas of the medical diagnosis domain gave rise to the PUFF System (for diagnosing pulmonary diseases), the CADUCEUS / INTERNIST (also for diagnosing bacterial infections), CASNET – Causal ASsociative NETwork (for diagnosing and suggesting treatments for the various types of glaucoma), to mention but a few, in which cases rules were just added to a knowledge-base by subject matter experts (domain experts) using text or graphical editors that were then integrated into the system shell. It must also be noted that Expert System shells are mostly useful for domains similar to the one the shell was originally written for.

Some other common examples of Expert System Shells or Tools include: CLIPS (for Rule-Based Expert Systems, built by NASA in 1985), JESS, AION-DS, Guru (USA), Personal Consultant plus (USA), Nexpert Object (developed by Neuron Data, California), Zeicon (developed by researchers at the University of Kent and Deakins University), Genesia (developed by Electricité de France, France), VP Expert (USA), EXPERTECH and EXPERTECH Xi Plus, KAS, KRITON, OPS (built by Forgy in 1977), ROSIE, ART, KEE (built by Intellicorp in 1983), LOOPS, Teiresias (built by Davis in 1973), HEARSAY III, AGE, RITA, AGNESS (developed by researchers at the University of Minnesota in 1986), Gestation Predictor [5] (1995) amongst others [17].

Furthermore, the Knowledge Acquisition bottleneck has remained through the years largely due to the fact that the Knowledge Engineers have remained the only interface between the Human Experts and the Knowledge Base of the Expert Systems at the development and engineering phases, coupled with the fact that the Knowledge Engineers in majority of (if not all) cases are novices in the domain areas which knowledge they're trying to Engineer (as illustrated in *figure 5* below).

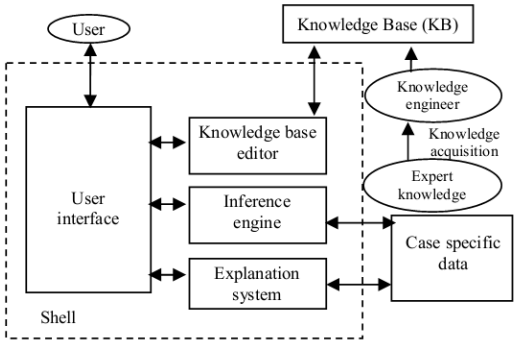

Figure 6: Expert System Architecture illustrating the current Knowledge Acquisition Process [19]

With this proposed new model for Knowledge Acquisition, Human Experts can be able to reuse these Expert System Shells and program (encode) their domain expertise directly into the Knowledge Base without the bottleneck associated with the Knowledge Engineers as the only interface between the Knowledge Base and the Human Expert at the development phases of the Expert System.

8. Suggestions for Further Studies

Research could be carried out in the area of the Knowledge Representation Language Formalisms of Expert System Shells in a bid to develop a higher level form of the Language, such that they are closer to Human Natural Language. Enhanced Data Type Abstractions, methods and/or classes could be developed and added to some of the existing High Level Programming Languages such that they could be more useful for core logic-based programming.

References

[1] Alan Garnham. *Artificial Intelligence: An Introduction*. London, United Kingdom. (1987).
[2] Bryan S. Todd. *An Introduction to Expert Systems*. Oxford University, Oxford, England. (1992).
[3] Cooper GF. *Current Research Directions in the Development of Expert Systems Based on Belief Networks*. Applied Stochastic Models and Data Analysis. (1989).
[4] Goodall A. *The Guide to Expert Systems*. Learned Information: Oxford, New Jersey. (1985).
[5] L.K. Woolery, D.S. Harr & D.L. Harr, "*An Expert System Shell as a Cost-Effective Tool for Predicting Preterm Birth Risk*." Journal of the American Medical Informatics Association, P (986). (1995).
[6] Lauritzen SL, Spiegelhalter DJ, "*Local Computation with Probabilities on Graphical Structures and Their Application to Expert Systems*." Journal of the Royal Statistical Society (Series B) (1988) (50) (ii) (157-224). (1988).
[7] MD Salim, A. Villavicencio, & M. A. Timmerman, "*A Method for Evaluating Expert System Shells for Classroom Instruction*." Journal of Industrial Technology Volume 19, Number 1. (2002).
[8] Maged N. Kamel Boulos "*Expert System Shells for Rapid Clinical Decision Support Module Development: An ESTA Demonstration of a Simple Rule-Based System for the Diagnosis of Vaginal Discharge*." Journal of Healthcare Informatics Research. 18(4): 252–258. (2012).
[9] Ovidiu S. Noran. *The Evolution of Expert Systems*. Griffith University, School of Computing and Information Technology, Queensland, Australia. (2003).
[10] PerfectLogic Corporation (2013), "*Artificial Intelligence and Expert Systems*." [Online]. Available: [http://www.perfectlogic.com/articles/AI/ExpertSystems/ExpertSystems.html]. Accessed: April, 2013.
[11] Peter Lucas (2007), "*Expert Systems*." An introductory scientific paper about knowledge-based expert systems, written for UNESCO. [Online]. Available: [http://www.cs.ru.nl/~peterl/eolss.pdf]. Accessed: April 2012.
[12] R. S. Engelmore & E. Feigenbaum, "*Knowledge-Based Systems in Japan*." Loyola College in collaboration with the Japanese Technology Evaluation Center. Maryland, USA. (1993).
[13] Robin, "*What Is Expert System?*" (2010). Articles on Artificial Intelligence. [Online]. Available: [http://intelligence.worldofcomputing.net/ai-branches/expert-systems.html]. Accessed: April, 2013.

[14] Sener, E. "*Using microcomputers and an expert system shell for soil classification*." Collegiate Microcomputer, 9 (1), 7-12. (1991).
[15] Stylianou, A. C., Madey, G. R., & Smith, R. D. "*Selection Criteria for Expert System Shells: A Socio-technical Framework*." Communications of the ACM, 35 (10), 30-48. (1992).
[16] Susan M. Land (1992). "*Expert System Design Shells: A Critical Analysis*." Penn State University. Instructional Technology Research Online. [Online]. Available: [http://www2.gsu.edu/~wwwitr/docs/esshells/], June, 2010. Accessed: April, 2013.
[17] "*The Development of Expert Systems Technologies*." [Online]. Available: [http://blue.cs.ksi.edu/dl/video/cis509/ch01aa.html]. Accessed: April, 2013.
[18] Trollip, S. & Lippert, R. "*Constructing knowledge bases: A promising instructional tool*." Journal of Computer-based Instruction 14(2), 44-48. (1987).
[19] Venus Samawi, Akram Mustafa & Abeer Ahmad "*Arabic Expert System Shell*." The International Arab Journal of Information Technology, Vol. 10, No. 1, January 2013. (2013).
[20] WISEGEEK. "*What Are Expert System Shells*?" [Online]. Available: [http://www.wisegeek.com/what-are-expert-system-shells.htm]. Accessed: April 2012.
[21] [Online].Available:[http://www.aiartificialintelligence.com/index.php?title=Expert_System]. Accessed: April 2012.
[22] [Online].Available:[http://www.wtec.org/loyola/kb/c3_s2.htm]. Accessed: April 2012.

Author Profile

Emmanuel C. Ogu received the B.Sc. degree in Computer Science (Technology) from Babcock University in 2011. He has been dynamically involved in the Training of ICT Professionals who has served in various areas of industry and academia since 2007. He shows special interest in researches relating to Artificial Intelligence and Neural Networks, Wireless Communications Technology and Systems, Information Security Systems and Cryptography. He is currently pursuing a Master of Science degree in Computer Science at Babcock University, Ilishan-Remo, Ogun State. Nigeria.