IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016

Model Interactions in a Domain Specific Modeling Language: An ICT Solution to Transformations in Design of Fluid Supply Systems

Igulu Kingsley Theophilus[1], Piah Z. Patrick[2], Japheth R. Bunakiye[3], Georgewill Moses Onengiye[4]

Dept of Computer Science, Ken Saro-Wiwa Polytechnic, Bori, Nigeria[1, 2, 4]

Dept. of Mathematics/Computer Science, Niger Delta University, Wilberforce Island, Nigeria[3]

Abstract: In domain specific modeling language development, model interactions and transformations play a crucial role. This paper introduces a new, syntax directed approach for the specification of modelinteractions for transformations within a domain specific modeling language. The technique is based on tokenization of the characteristic values and attributes of possible component elements of a fluid supply system. To demonstrate the mapping, expression parse trees and streams of tokens are recursively specified as syntax directed definitions in a context free grammar. With each grammar symbol, a set of attributes are associated, and with each production, a set of semantic rules for mapping values of the attributes corresponding to a typical fluid supply system physical design and modeling parameters are associated. The main contributions are the specifications that will aid the design of a domain specific modeling language for the representation of transformations and translation scheme execution engine for the target domain of fluid supply systems, and another is a possible pathway for the implementation of the semantics of the language with a syntax directed scheme that consists of a grammar and the set semantic rules.

Keywords: Model Driven Engineering, Domain Specific Modeling, Values and Attributes, Context Free Grammar, Transmission Pipelines, Joints and Fittings.

1. INTRODUCTION

Domain Specific Modeling (DSM) provides solutions for computing and software platform complexities. It permits software engineers and related scholars in the field of Model Driven Engineering Technologies (MDE) [1] to hide the details of the platforms by raising the level of abstraction on which applications are built. The basic concepts of the platform frameworks are represented as the available kinds of objects in a new, domain-specific modeling language (DSML) [6]. The main focus of this paper is that using these high-level concepts in a formal modeling language structure, these models can be processed to generate workable transformations that provides best advantage for design of fluid supply systems [7]. A platform that could foster design activities, with fluid supply system design as an integral part of model interactions to which the work can be applicable. Naturally [3], domain-specific languages have capabilities for extensions, making their use easier and more consistent. It does therefore make some sense to present domain specific features with unique functionalities for use by domain experts, rather than wasting time writing all code character by character in order to achieve results. Usually the first steps proceeds with specifying the modeling language, and, fixing the abstractions; this capability creates model interaction in such a way that transfer of information is possible within the set conditions in the modeling language system [16]. One basic relevance in the ease of design transformations is the issue of interaction between models, interactions in the way of concepts devoid of possible parametric constraints as applicable with conventional modeling systems [2]. Interactions that can produce other complete models with noticeable properties relative to a given set of concerns in the fluid supply domains that captures accurately and concisely all of its interpretation and design intent for the specific problems and solutions [5]. The transformation impact in this context refer to the DSML software engineering methodology used to engineer model interaction in order to create new objects that encapsulates and relates the details pertinent to the viewpoints of the domain experts. Transformations in this pattern will raise productivity levels per man hour [8]. The remainder of this paper is outlined as follows: Section 1 of this report highlights an overview of domain specific modeling; section 2 gives a brief of related literature in the field of model driven engineering and model transformations. Section 3 of the paper gives an analysis of the domain of pipeline systems that transmit fluid from source to destinations. Sections 4 to 5 presented a detailed specification of attributes and values of components commonly found in fluid supply systems, and the possible interactions that are feasible in a domain specific modeling language to bring about transformations to meet the needs of experts in the industry. Section 6 gives the conclusion and future work.

2. BACKGROUND AND RELATED WORK

2.1 Models and Model Driven Engineering

Model -Driven Engineering (MDE) is a kind of software development approach, where models are seen as first class entities [4]. In MDE everything is a model, a model conforms to another model and model transformations takes models and produces models. Models are human concepts that explain systems in the real world, they are simply outcomes from abstractions, which means all domain specific modeling languages under MDE could provide domain experts some level of consciousness of an abstract model before bringing it forth through transformations in the modeling system internal mechanisms and logic [9].

The model must represent concepts in the specific domain, the concepts has to come from the elements and building blocks of the library frameworks in order to set the abstractions and grouped to match the domain experts' abstract model of the problem domain. These salient characteristics are essentially embodied in the domain model. Talking about the domain of fluid supply systems, getting the concepts for a proper design that depicts stakeholders' viewpoints is purely an engineering design process. Engineers combine quite a number of design methods for solving most of their modeling problems. One of such exploration is the graphics or computer model. However, the use of computer models for engineering design practice is currently not seen to be productive in the MDE community; much more emphasis is centred on modeling these models to products that satisfies varied design intent.

As much as there are lots of software platforms suited for modeling, they also portend a lot of shortcomings; users are often limited by their knowledge of the software or by problems solvable by it. These deficiencies tend to make engineering design standards as merely applicable to the creation of physical objects or, perhaps, software. A more appropriate approach is the application of a DSML as layers of reusable software to solving some such design issues pertaining to fluid transmission. The selling point is the ability for models to interact via their attributes and values within the internal dynamics of the DSML.

2.2 Related Work

Regarding the kind of syntax definitions and requirements, domain specific languages are typical resource for model transformations. K. Hölldobler, B. Rumpe, and I. Weisemöller[17] presents a process that allows to systematically derive a textual domain specific transformation language from the grammar of a given textual modeling language. They applied a systematic derivation of the semantics to UML class diagrams to obtain a domain-specific transformation language called CDTrans. This was demonstrated by incorporating familiar concrete syntax of the UML class diagrams and extending them with a few transformation operators. The present specifications in this paper are not restricted to just the UML software process but can be applied to any specific domain in the fluid transmission industry. Bernhard Rumpe and Ingo Weisem¨oller[18] discuss a domain specific modeling method for modeling parts of the system under development in a problem-oriented notation that is well-known in the respective domain. This technique is applicable to domain specific modeling languages that are often accompanied the desire to transform its instances.

As such, it complements our technique, which an approach is tied to the tokenized instances of the language creation. Similarly, Tom Mens, Krzysztof Czarnecki, and Pieter Van Gorp[19] use a Language Engineering taxonomy of model transformation executions. The focus of their work is on helping developers in deciding which model transformation approach is best suited to deal with a particular problem. JochenKüster's [20] research uses the model driven architecture approach for iterative refinement of models by model transformations. Conceptually, this work is closely related to ours; automation of recurring tasks can often be achieved by model transformations.

3. FLUID SUPPLY SYSTEMS

3.1 Design Considerations

Physically, fluid supply systems are virtually pipelines. Pipelines are the most common means of transporting fluids [21]. Like any other flowline, small sections of pipeline are not easily removed for maintenance and consequently great care is taken to prevent problems arising in the first place. A pipeline is extremely expensive to lay, so when designing a pipeline, the engineer must consider the volume and the physical and chemical properties of the fluid, the nature of the environment through which the pipeline is going to traverse, and more specifically the diameter, length, size and type of pipe and other components.

Also applicable are calculations in the design of oil and gas pipelines, certain calculations pertinent to the smooth running of the pipeline has to be done. In most pipeline calculations, assumptions must be made initially. For instance, as exemplified in figure 1[21] a line size may be assumed in order to determine maximum operating pressure and the pressure drop in a given length of pipe for a given flow volume. Assumptions on valves, pumps joints and fittings also contribute to overall pipeline system maximum performance and integrity. Once a domain specific language system with familiar pipeline engineering notations are available for domain experts to understands and design a pipeline, it should not be difficult to design and built any other of any type. Since standard pipe grades, sizes and weights are normally used, and maximum operating conditions specified, several formulae in terms of differing scenarios can be used to calculate the flow of oil and gas in a pipeline. These scenarios account for the effects of the attributes in the designs.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016

Fig. 1. A Typical Pipeline (Source: www.cobbfendley.com Transmission Pipeline Engineering Services)

3.2 The Model Driven Pipeline Build Process

Pipeline build process involves structural design in relation to loading and stresses, and determining route selection, i.e. the origin and destination of the pipeline. Other factors include the approximate length of the pipeline, the product to be transported, diameter and type of pipe used, hydraulic factors such as type of flows expected in a pipeline, approximate capital cost and running expenses [10]. Fluid supply pipeline projects are large scale multi-disciplinary activities that stems from conceptual capture through designs, which involves the investment of large amounts of cash and other resources [21]. Because of this, and the fact that productivity is the order of modern day large scale pipeline projects the development and implementation of a pipeline project involves so much design efforts. It is characterized with numerous compliance issues even after design is completed and the project kicked off. With model driven design, the model is the heart of the design, and the methodology of manipulating this model to ease the project execution for users becomes paramount. Creating and specifying the domain model in a metamodel of relationships and attributes will result in a modeling language [11]. The modeling language, which is specific to the domain of fluid supply engineering, will be capable of allowing the experts to easily get specifications correct on time and at very minimal costs instead of always returning to conformal forms and related issues (Eric and Oliver, 2012).

4. DETAILED METHOD STATEMENT

4.1 The DSM Approach

The DSM approach is a software engineering technique for producing layers of reusable software platformsto perform and managemodeling. DSMshelves the user from syntax and programming complexities associated with most conventionalsoftware platforms. DSM utilizes a combination of the capabilities of an application model, a solution model, and then a knowledge base of the repositories of associated domain concepts. With these capabilities embodied in a rule processing domain specific modeling language, all aspects of design criteria reports and fabrication operations of any fluid supply systemcan be generated, providing operators with information on the systems current requirements for model interactions and possible orientations.

Usually the application model is the user interface family of domain notations, which forms the basis for solutions to the design of the typical pipeline system for the supply of fluids. It is the interactive layer that provides the pipeline design factors, fundamental elements, and parameters. Included also are dimensional criteria, instruments criteria, support positioning, location classification and use of standards [13].
Interacting with the system for a design operation requires some planning. Planning to learn about the elements considered in the design, code, factors, legislation, material selection, diameter selections (internal and external) considerations, and management is an essential step.Training and deep rooted knowledge about increased through put through careful selection of design criteria is also a necessary step. Major contributions on the part of the user at the application layer are knowledge on pipeline systems considered from a design point of view; the length, diameter, and loop placements precisely tailored to suit new load profiles. Increases in throughput of designed pipelines are readily accomplished if the diameter and length of the loops are sufficient. Pipelines can also be looped in stages to suit increasing demands on a defined timetable. The length of time required to loop a whole pipeline or pipeline sections is dependent on the amount of pipe and other components to be installed [40].

4.2 Model Specifications and Interaction

It is important to note that models in a DSML are constructed using concepts that represent targets within the application domain. Models interacting in a language system, which suffices the design of a typical pipeline system that supplies fluid must represent concepts relating to any fluid pipelines. Specifying the interactions of such models therefore means the creation of the language metamodel. These models, which are basically the instances of the language metamodel are in the form of any relevant physical components such as pipes, fittings, joints, anchors hangers, stanchion, bolts and nuts, and gaskets and other pressure instruments like pumps, valves etc. Basic interactive component elements include joints in the form of joint types, joint dimensions, fittings in the form of fitting types, fitting dimensions, and the points of interaction for all of pipe (p), fittings, and joints in the order of (f.t.d.p for fittings; j.t.d.p for joints), and other components in the stream (See Fig 2).

As far as a design scenario is explicitly identified through stakeholder relations with the language concrete syntax elements, it means then that internal communication among these models in their grammar units have enforced and made some possible interactions. Notable conventional applications in language design adopted here is the integrated semantic module parser component in the language meta model.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016

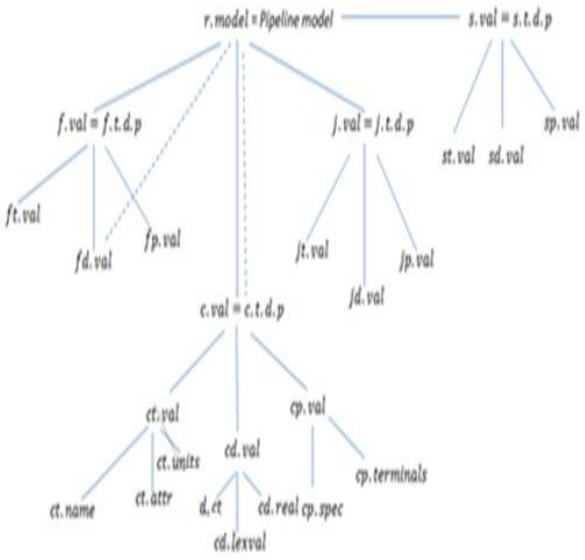

Fig. 2. Model Specifications

For proper orientation and adequate model interactions in the internal mechanism, the semantic module encompasses mappings through syntax directed definitions that consists of the grammar and the set of semantic rules [12]. As shown in figure 2 the syntactic elements are specified as symbol substitutions for the major objects in the pipeline model that can be recursively performed to generate new modeling sequences and to keep track of domain specific relevant information [17]. The information is tagged with the pipeline component attributes (attr) and values(val),which can be transferred into the instruction sequence in the language construct.

4.3 Design Considerations

Oil and gas pipelines design scenarios, either onshore or offshore are usually set off by increasingly complex challenges in the exploration and development of energy resources. Successful execution of the design systems therefore requires innovation and creativity, and the passion to deliver must consider many variables: safety, technical, financial, environmental, regulatory, logistics and culture [13]. Companies engaged in fluid transmission activities, for example oil and gas pipeline companies prefer to operate their systems as close to full capacity as possible to maximize their revenues. This is actually one development strategy where a design project in the domain integrates storage capacity into the pipeline network design so as to increase average utilization rates. This integration will then showcase designs that are capable of balancing flow levels by moving products to and from storage facilities. Major pipeline design circumstances and scenarios considered in our system are restricted to interactions of models specific to the domain of consideration, which is in this context a pipeline system that supplies fluids to and from storage facilities. Necessary inclusions are expansion parameters, control capacities, design dimensions and the addition of looping; which means adding a parallel pipeline along a segment of pipeline, addition of features for the building of an entirely new pipeline, and features for upgrading and expanding facilities, such as compressor stations, along an existing route. This methodology ensures some accuracy of calculations of the physical components attributes and values. These calculations yield significant system advantages because typical pipeline that transports oil and gas is expected to maintain uniform standards. To achieve this, a domain specific modeling language, which can enable model interactions to be able to bring to bear required transformations for the design of fluid supply systems is now becoming indispensable [14].

5. MODELING INTERPRETATIONS

5.1 Tokenization

In any typical physical fluid supply system, lots of models such as pipes, fittings, joints, anchors hangers, stanchion, bolts and nuts, and gaskets and other pressure instruments like pumps, valves are involved in varied connections and dimensions. Interactions of these components in their values and attributes is observed to be very feasible in a language with a stream of tokens; having the resultant effects of additional models or executable codes generation. Thevalues and attributes earlier exemplified in figure 2 represents the fluid supply system possible components characteristics. They are the stream of tokens that make up the language grammar that is segmented into its word units [4] as shown in figure 3. With each grammar symbol, and with each production, a set of attributes and a set of semantic rules are associated to corresponding physical design and modelling parameters of an entire fluid supply system. The entire lexemes in the token definitions are therefore syntax directed and consists of the grammar and the set semantic rules [6]. In line with domain specific modeling, the DSML platform needs to keep track of all these domain specific relevant information by tagging this information with attributes and associating the attributes with the language metamodel [11].

$$Tokens: \rightarrow f = ft * fd * fp$$

1. $The identifier - fittings(f)$
2. $The assignmentsymbol - (=)$
3. $The identifier - fittingtype(ft)$
4. $The multoperator - (*)$
5. $The identifier - fittingdimension(fd)$
6. $The identifier - fittingpoint(fp)$

$$Tokens: \rightarrow j = jt * jd * jp$$

1. $The identifier - pipingjoint(j)$
2. $The assignmentsymbol - (=)$
3. $The identifier - jointtype(jt)$
4. $The multoperator - (*)$
5. $The identifier - jointdimension(jd)$
6. $The identifier - jointpoint(jp)$

DOI 10.17148/IJARCCE.2016.5685

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016

$f \rightarrow ft \mid fd \mid fp$ $\quad$ $j \rightarrow jt \mid jd \mid jp$

$ft \rightarrow expr \mid term \mid factor$ $\quad$ $jt \rightarrow expr \mid term \mid factor$

$fd \rightarrow list \mid digit \mid expr$ $\quad$ $jd \rightarrow list \mid digit \mid expr$

$fp \rightarrow expr = string + string \mid string - string$ $\quad$ $jp \rightarrow expr = string + string \mid string - string$

Fig. 3.Stream of Tokens

5.2 Expression Parse Tree Interpretation

The stream of tokens earlier exemplified in figure 3 makes up the language grammar that is segmented into its word units. These units are identified and further decomposed into parser processing. Mapping the core concepts produces the grammar rules. The production rules are basically specifying the symbol substitutions for the major objects in the fluid supply system that can be recursively performed to generate new modeling sequences [15]. Adopting this assertion to the syntax-directed definition in the DSML, the set of attributes (attr) and values (val) are associated with each grammar symbol, and the set of semantic rules with each production are all depicted in the expression parse trees shown in figure 4. The hierarchical presentation of the parse trees is indicative of the fact that the grammar productions are expressed by recursive rules [9]. The entire structure is a collection of fluid supply system components context free grammar comprising all the correspondingtokens for processing. These grammar phrases express the identifiers as generic collections that are iterated to produce the desired expression tree, taking into account the components build parameters and the associated grammar for each of the components. The resultant effect of the internal working mechanism of the DSML is an interpreter program running on the target platform that loads the program, and then acts on it.

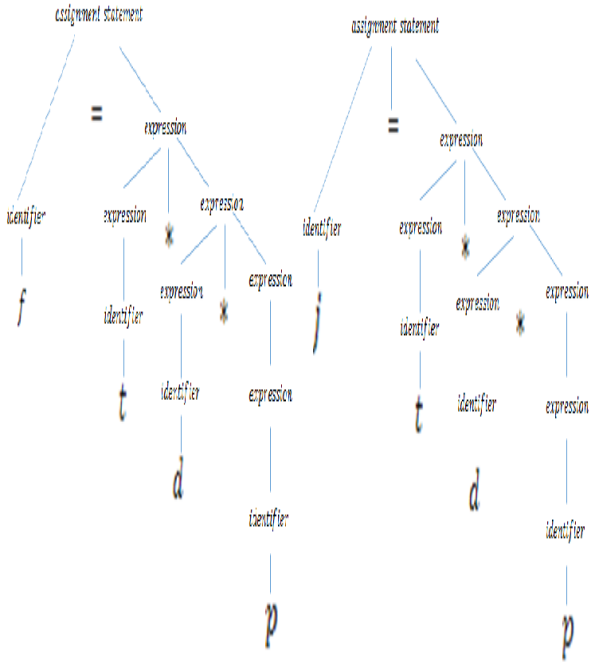

Fig. 4: Expression Parse Trees

6. CONCLUSION AND FUTURE WORK

A philosophy of modeling based on the fundamental elements of a fluid supply system is presented. The processes involve specifying the grammar elements of the underlying physical component with the intention of highlighting the essential steps for interactions in a domain specific modeling language. This modeling language has the basic features of a syntax directed definition for transformations that allows flexible development and specification. The resulting tool design and development is the next very important step in the future work. The flexibility of the language is based on tokenization of the characteristic values and attributes of possible component elements of a fluid supply system. The main contributions are the specifications that will assist in the design of the domain specific modeling language for the representationsand the possible pathway for the implementation of the semantics of the language.

REFERENCES

[1] Thomas Stahl and Markus V□olter. Model-Driven Software Development. Wiley,2006.
[2] Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham,Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Je_ries,et al. Manifesto for agile software development. 2001.
[3] B. Vanhooff and Y. Berbers. Supporting Modular TransformationUnits with Precise Transformation Traceability Metadata. In ECMDA-TW: Traceability Workshop, at Eu-ropean Conference on Model Driven Architecture, Nuremberg,Germany, November 2005.
4] Yuefeng Zhang and Shailesh Patel. Agile model-driven development in practice.IEEE Software, 28(2):84{91, 2011.
[5] Vinay Kulkarni, Souvik Barat, and UdayRamteerthkar. Early experience withagile methodology in a model-driven approach. In 14th Int. Conf. on Model DrivenEngineering Languages and Systems, pages 578{590, 2011.
[6] Lorenzo Bettini. Implementing Domain-Specific Languages with Xtext and Xtend.Packt Publishing Ltd., 2013.
[7] Hamid Bagheri and Kevin J. Sullivan. Bottom-up model-driven development. In35th Int. Conf. on Software Engineering, pages 1221{1224. IEEE/ACM, 2013.
[8] Jernej Novak, Andrej Krajnc, and RokZontar. Taxonomy of static code analysistools. In MIPRO, 2010 Proc. of the 33rd Int. Conv., pages 418{422. IEEE, 2010.
[9] TimoKehrer, UdoKelter, and Gabriele Taentzer. Consistency-Preserving EditScripts in Model Versioning. In 28th IEEE/ACM Int. Conference on AutomatedSoftware Engineering, pages 191{201. IEEE, 2013.
[10] Ste_enVaupel, Gabriele Taentzer, Jan Peer Harries, Raphael Stroh, Ren_eGerlach,and Michael Guckert. Model-driven development of mobile applications allowingrole-driven variants. In 17th Int. Conf. on Model-Driven Engineering Languagesand Systems, pages 1{17, 2014.
[11] P. Stevens. Bidirectional model transformations in QVT: Semantic issues and openquestions. In Proc. of the 10 Int. Conf. on Model Driven Engineering Languages andSystems, Lecture Notes in Computer Science, pages 1–14. Springer, 2007a.
[12] Lennart CL Kats, EelcoVisser, and Guido Wachsmuth. DSL Engineering – Designing,Implementing and Using Domain-Specific Languages. dslbook. org, 2013.
[13] Bauer, F.L., Ehler, H., Horsch, A., M¨oller, B., Partsch, H., Paukner, O., Pepper, P.:The Munich Project CIP, Volume II: The Program Transformation System CIP-S,Lecture Notes in Computer Science, vol. 292. Springer (1987)
[14] B. Rumpe and I. Weisemöller, "A Domain Specific TransformationLanguage," in Workshop on Models and Evolution (ME), 2011.

[15] M. Schmidt, "Transformations of UML 2 Models Using Concrete SyntaxPatterns," in Rapid Integration of Software Engineering Techniques, ser.Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007,vol. 4401, pp. 130–143.
[16] R. Grønmo and B. Møller-Pedersen, "Concrete syntax-based graphtransformation," 2009, research Report 389.
[17] K. Hölldobler, B. Rumpe, and I. Weisemöller: Systematically Deriving Domain-Specific Transformation Languages. In: Conference on Model Driven Engineering Languages and Systems (MODELS), pp. 136-145, Ottawa, Canada, ACM New York/IEEE Computer Society, 2015 www.se-rwth.de/publications
[18] Bernhard Rumpe and Ingo Weisem¨oller. A Domain Specific Transformation Language Software Engineering RWTH Aachen University, Germany http://www.se-rwth.de/
[19] Tom Mens, Krzysztof Czarnecki, and Pieter Van Gorp: A Taxonomy of Model Transformations, Software Engineering Lab, Dagstuhl Seminar Proceedings 04101 http://drops.dagstuhl.de/ opus/volltexte/2005/11
[20] JochenKüster: Model-Driven Software Engineering Model Transformations IBM Research – Zurich (jku@zurich.ibm.com) MDSE 2011Motivation for Model Transformations
[21] CobbFendleyTransmission Pipeline Engineering ServicesDallas County, Texas 2014 Mike Diamantini13430 Northwest Fwy., Suite 1100Houston, Texas 77040 www.cobbfendley.com

BIOGRAPHIES

Igulu Kingsley Theophilusis a lecturer in the Dept. Of Computer Science, Ken Saro-Wiwa Polytechnic, Bori, Nigeria. He is a very sound computer scientist in the areas of Machine Learning, Agent Oriented Programing and Model Driven Engineering Software Technologies

Piah, Z. Patrick is a Principal Lecturer in the Dept. Of Computer Science, Ken Saro-Wiwa Polytechnic,Bori, Nigeria. He is very experienced and specialized in Computational Geometry and Artificial Intelligence

Japheth R. Bunakiye is a senior lecturer in the Dept. of Mathematics/Computer Science, Niger Delta University, Wilberforce Island, Nigeria. His areas of specialization include Model Driven Engineering Technologies, Software Requirements Engineering, and Methodologies

O.M.D. Georgewillis a chief lecturer in the Dept. Of Computer Science, Ken Saro-Wiwa Polytechnic, Bori, Nigeria. He is specialized in Artificial Intelligence, Fuzzy Logic and Information Systems Concepts

DOI 10.17148/IJARCCE.2016.5685