World Academy of Science, Engineering and Technology
International Journal of Cognitive and Language Sciences
Vol:13, No:2, 2019

A Domain Specific Modeling Language Semantic Model for Artefact Orientation

Bunakiye R. Japheth, Ogude U. Cyril

Abstract**—**Since the process of transforming user requirements to modeling constructs are not very well supported by domain-specific frameworks, it became necessary to integrate domain requirements with the specific architectures to achieve an integrated customizable solutions space via artifact orientation. Domain-specific modeling language specifications of model-driven engineering technologies focus more on requirements within a particular domain, which can be tailored to aid the domain expert in expressing domain concepts effectively. Modeling processes through domain-specific language formalisms are highly volatile due to dependencies on domain concepts or used process models. A capable solution is given by artifact orientation that stresses on the results rather than expressing a strict dependence on complicated platforms for model creation and development. Based on this premise, domain-specific methods for producing artifacts without having to take into account the complexity and variability of platforms for model definitions can be integrated to support customizable development. In this paper, we discuss methods for the integration capabilities and necessities within a common structure and semantics that contribute a metamodel for artifact-orientation, which leads to a reusable software layer with concrete syntax capable of determining design intents from domain expert. These concepts forming the language formalism are established from models explained within the oil and gas pipelines industry.

Keywords**—**Control process, metrics of engineering, structured abstraction, semantic model.

I. Introduction

DOMAIN specific modelling languages (DSMLs) [8] are usually small, declarative languages; they are executable specification languages or simply any artificial language that can be used to express information or knowledge systems in a structure that is defined by a consistent set of rules. The rules are used for the interpretation of the meaning of components in the structure. Established development processes of DSMLs we have embraced require binding the syntactical concepts of the problem domain with the semantics of a solution domain [9]. This is accomplished by raising the level of abstraction and then focusing on domain concepts during design. But the challenge has been the lack of a semantic definition within the context of the defining metamodel. Most of them capture abstract syntax and constraints, but the issue of domain semantics is usually embedded in one or more code generators [10]. Some of them, for example DSML-based tools, such as Simulink has been slowed down by the lack of complete and formal semantic specifications, and has made the precise understanding of their syntax and semantics difficult [2]. Often users run a high risk of being constrained in a particular tool chain. This problem has negative impact on reusability of DSMLs, because a well-made DSML [6] captures the concepts, relationships, integrity constraints, and semantics of the application domain and allows users to program declaratively through model construction. We have therefore incorporated in this work a semantic module to alleviate this challenge. The aim is to offer designers modelling concepts and notations that are tailored to characteristics of their application domain [10]. The shift is a move toward an infrastructure for DSML design that integrates formal methods with practical pipeline engineering principles [11]. Artifacts in model-based design and development, however, are seen as a structured abstraction of modeling elements used as input, output, or as transitional results of a modeling process [3]. An artifact can feature particular structural and behaviour properties when precisely described using formal modeling concepts. Such descriptions will enable the combination of different and clear responsibilities to support a progress control for the production of artifacts. The possibility of the entire description and control process is enhanced through a semantic model within the process. Furthermore, for artifacts to be defined for positioning, the development environment must be capable of abstracting from complex methods or tools [4]. However, a primary step towards such definition and use consist domain component relations for a particular orientation process, in our case for the oil and gas pipelines domain, is a pipeline model. This guarantees that all coordinating component elements in a typical modeling scenario must have distinguishing vocabulary of what will be exchanged within this semantic model [12]. Once the semantic model is defined, it is requisite for components and all their attributes and values to start working based on the mechanics of the constraints in it. This achieves finally a seamless integration of a semantic module in a domain specific language to ease design, code generation, and quality assurance in software development [5].

II. Related Work

In terms of supporting stakeholder competing modelling intents in the oil and gas pipelines industry, a software development effort aimed at collaboration and easy access to view points are crucial for successful artefact productions. Besides the effort required for understanding a computer aided design system or other programming platforms in defining custom models, the inclusion of a semantic model in a DSML formalism as in our case will provide any functionality for

B. R. Japheth is with the department of Computer Science, Edo University Iyamho, Edo State, Nigeria (corresponding author, phone: +234 (0)8061324564; e-mail: japheth.bunakiye@edouniversity.edu.ng).

O. U. Cyril is with the department of Computer Science, University of Lagos, Akoka, Lagos, Nigeria (e-mail: uogude@unilag.edu.ng).

applying and executing models satisfying stakeholders design intents. Jie Hu et al. [13] in their approach presented a semantic model for academic social network analysis. The social network is a system of common players of some conformity. What they did was to integrate a semantic model in their analysis program to properly and effectively represent, manage and use various social networks. It was a semantic model that can naturally represent various academic social networks especially various complex semantic relationships among social actors. Focusing on graphical user interface (GUI) programming, Neelakantan and Nick [14] described in the software metrics a semantic model as an ultra-metric structure that enforces causality restrictions and allows well-founded recursive definitions. Their aim is to capture the arbitrariness of user input by allowing a user to decide the stream of clicks sends to the program so that the non-linear part of the language is used for writing reactive stream-processing functions, whilst the linear sublanguage naturally captures the generativity and usage constraints on the various linear objects in GUI. Our semantic model actually gets beyond the GUI and moved on to manipulating generation of artefacts. A Semantic-Web Approach for Modeling Computing Infrastructures was developed by Mattijs et al. [15]. Their main contribution is the Infrastructure and Network Description Language (INDL) ontology. The aim of INDL is to provide technology independent descriptions of computing infrastructures, including the physical resources as well as their network infrastructure. Unlike INDL, our DSML for oil and gas pipelines was not based on multi-processing environments rather it focused on a collaborative communication pathway for proper orientation of components. Méndez et al. [7] in their work explained a metamodel for artefact-orientation based on fundamentals and lessons learned in requirements engineering. They looked at repetitive activities complicating a standardized RE process such as dependencies and used process models. They then provided a promising solution through artefact orientation that emphasizes the results rather than dictating a strict development process. This approach compliments our own especially placing much emphasis on the results rather than relying on complicated design or programming platforms.

III. Method and Materials

The domain of consideration is oil and gas transmission pipelines. A pipeline system dedicated for transmission of oil and gas from wells to tanks for storage or to refineries for processing. Such pipeline system includes pipe sections joined with fittings and other supports features such as flanges, fittings, bolting, gaskets, valves, hangers and the pressure containing portions of pipeline components. The components of the pipeline systems, which are here referred to as the pipeline model are represented as AutoCAD graphics objects that depict the typical pipeline fundamentals, materials and joints [1]. These pipeline models, as shown in Fig. 1, actually form the instance of the semantic model.

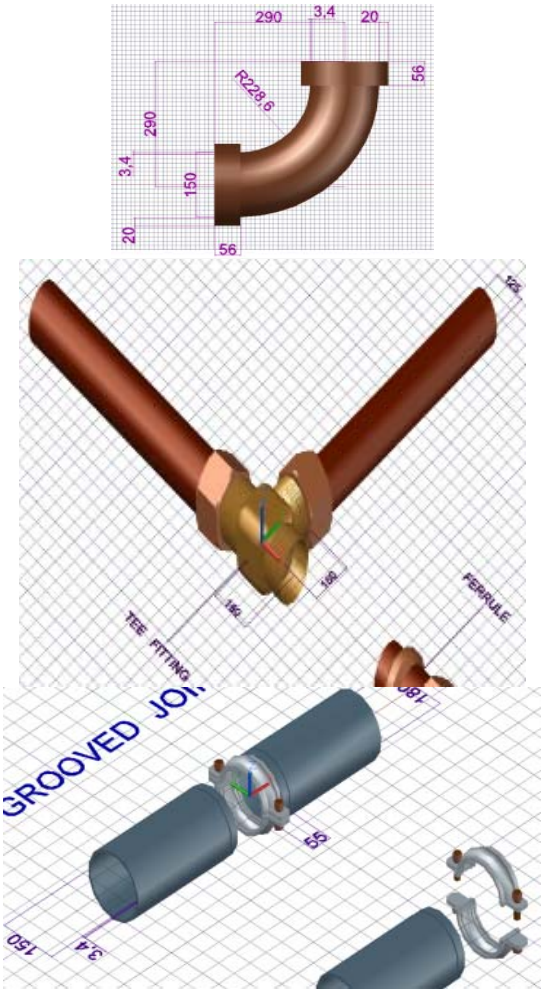

Fig. 1 Pipeline Model

Now expressing these models in a language definition specific to the mechanisms of oil and gas pipeline systems, internal communication among collaborative sub-systems are created in a domain model. The domain model contains the semantic model incorporating the abstract syntax, semantics and the concrete syntax definitions of the language that captures the metrics of the pipeline engineering field; the stakeholder can then interaction with the system through a guided interface and the system can then match these inputs with the parsing grammar to produce desired artefacts.

A. Semantic Definitions

The domain model incorporates all of the communication abstractions and their relationships. The semantics units within the abstraction layers are precisely defined to capture concurrency so that the systems should meet the requirements of the operating environment. The semantics are defined in the sense of oil and gas pipeline product line model (PPM). A product (P) of the feature model holds information regarding a set of leaves. The model (M) holding values of products as a subset of primitive nodes, and the constraints (C) are the rules guiding the specific definitions.

$$APPMisatriple(P, M, C)$$

where:

$$P = \{p_1, p_2, p_3, \ldots\ldots, p_e \in P\} isthesetofleavesP = p(F)$$
$$M = m \in MisthesubsetofthenodesoftheFD: M = pN$$
$$C = \phi \in \Phi; arethesatisfyableconstraints$$

$$Definition\ 1$$
$$Theproductlinewithasetoffeature$$
$$leafs(F)ofthePPMisavalidmodelm \in Miff:$$
$$(a) Theconceptisinthemodel: r \in m$$
$$(b) i. Ifaxor - nodeisinthemodel, exactlyoneofits$$
$$subfeaturesis: q \in m$$
$$\Rightarrow \exists! n. q \rightarrow n \wedge n' \in m$$
$$ii. Ifaand - nodeisinthemodel, allitssubfeaturesare: k \in m$$
$$\Rightarrow \forall n. (k \rightarrow n \Rightarrow n' \in m)$$

$$(c) i. Themodelmustsatisfytheconstraintset: \forall \phi \in \Phi, m \vDash \phi$$
$$where:$$
$$m \vDash n_1 mutexn_2 \rightarrow n_1 and n_2 \notin m;$$
$$m \vDash n_1 requiresn_2 \rightarrow n_1 n_2 \in m$$
$$ii. Satisfythetextualconstraints: \forall \phi \in \Phi, m \vDash \phi,$$
$$wherem \vDash \phi_1 \rightarrow p_1 \in m \Rightarrow p_2 \in mistrue$$
$$iii. Satisfythegraphicalconstraints: \forall (n_1 * n_2) \in C, * (n_1 \in m, n_2 \in m) istrue$$

$$Definition 2$$
$$TheproductlinePwithasetoffeatureleafs$$
$$(F) ofthePPMisthesetofleavesP = p(F)\ iff:$$

$$(a) Theproductlineisasetofproducts: p(P) = p(p(F)).$$

$$(b) Theproductlineistheproductsofthe$$
$$PPMsvalidmodels:$$
$$suchthat[\![p]\!] = \{m \cap F \setminus m \vDash p\}$$

$$(c) Aproductkisasetofprimitivenodes: k \in p(P).$$

B. Semantic Model Specifications

The operational mode is hinged on the description of the grammar content of the input parameters in the formal notations. Which means all aspects of a typical design scenario, criteria reports and fabrication operations can be generated, providing operators with information on the pipelines current installation requirements. Fig. 2 is a hypothetical pipeline build event that is carried out in real life. It is a problem solving venture that minimizes the complexities encountered in the engineering design workplace.

Following the necessary structural framework that must be put in place for DSML to implement its core operations, Fig. 3 is showing the creation of the grammar fragments in BNF notation for defining the various pipeline build metrics of the language.

```
Define PpelineBuild Parameters
build_pipe
{
component = (); (double)
fitting = (); (double)
joint = (); (double)
support = (); (double)
angle = (); (double)
size = (); (diameter_inner(float))
(diameter_outer(float))
(width(int))
(thickness(int))

units = (); (int)
point_x = (); (int)
point_y = (); (int)
point_z = (); (int)
}}pipe_join.method.thispipe.p(0,1)
```

Fig. 2 Fragments of Pipeline Build Grammar

```
Build Pipeline Eq Build
Size Eq Dimension
Builder *
End Id
Pipeline Build metrics ::
= shape|direction|component|fittings|joint|support
Eq ::= " = "
Builder ::= Builder Id
Point Eq Position
End Id
Position ::= origin|attributes|length|slope|terminal
Definitions ::= Definition *
Definition ::= Define Id Shape ;
Width Eq Number; Thickness Eq Number
```

Fig. 3 Formal Notations in BNF

The language encompasses in its domain model logic sound underlying engineering principles, which illustrates how they are linked to produce a total life cycle approach to pipelines system design and operation. In the domain model is the semantic model subset consisting of the classes of the events and their relationships with a focus on the user's perspectives. An example of a typical event pertaining to user's perspective is given in Fig. 4.

```
event
name: elbowJoint
code: PipeBiuld
end

state: active
elbowJoint → WaitingForParameters
end

state: join.this.elbow
translate → target
target: name → join.this.elbow
trigger: elbowJoint
end
```

Fig. 4 Semantic Model (User's View)

As knowledge changes, the semantic model itself can change so as to ensure physical components continue to do

what the users want them to do and then produce clear design specifications for pipeline physical assets such as pipes, valves, active equipment (pumps, compressors, etc.,), insulation and supports. Common and differing functions are also abstracted and represented so that specific design scenarios can be evolved as an adaptation or refinement of the pipeline model.

IV. Discussion of Results

The underlying engineering principles in the domain model logic and their relationships are the user centered composition rules of the semantic model comprising the events handler. The domain model which represents the relationships and classes of the core features of the application domain captures all the semantic behaviors of these features in the form of concepts and specifications. In order to evaluate these semantic behaviors as the overall performance of the modelling system, a text template transformation, as shown in Fig. 5, is performed on the build tab to trigger a transformation of the class templates to the target platform (i.e. the UI text editor platform) that loads the DSML program.

The resultant effect of the internal working mechanism is that the DSML interpreter program (i.e. the event handler DSML scripts) runs as a layer over the semantic model (i.e. the library framework) and populates it. Illustrated in Fig. 6 is the semantic representation, it is the in-memory object model with the state of events of the same input metrics that our DSML describes.

During parsing, which is the next stage, the interpreter program running on the target platform acts on the events states and comes up with a result oriented abstract syntax tree (AST). In the context of our DSML, the semantic model representation has clearly indicated the data binding process to be an object binder that specifies the event states. The events become more vivid in the form of text inputs from the UI, also, the template transformation has made it possible for the functionality of a new text editing platform i.e. the creation of the editor. What happens is that the components container binds the data source from the internal representations to the DSML model, particularly to the root node of the model. Shown in Figs. 7 (a) and (b) is code snippet for the data binding actions that performs the user control in the editor with a resultant example of a typical modelling action.

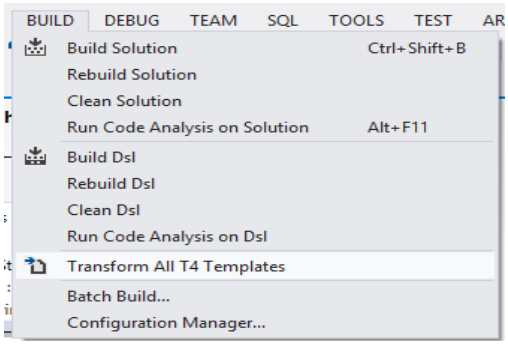

Fig. 5 Template Transformation

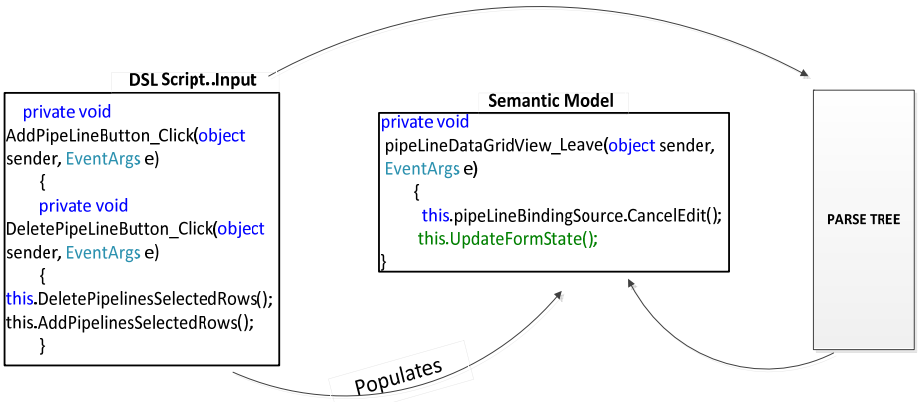

Fig. 6 Semantic Model Representation

```
public partial class PipeLineControl : UserControl
    {
        public PipeLineControl()
        {
            InitializeComponent();
        this.pipeLineDataGridView.KeyDown += new
System.Windows.Form.KeyEventHandler(pipeLineDataGridView_KeyDown);
this.fittingsDataGridView.KeyDown += new
System.Windows.Form.KeyEventHandler(fittingsDataGridView_KeyDown);
this.componentsDataGridView.KeyDown += new
System.Windows.Form.KeyEventHandler(componentsDataGridView_KeyDown);
this.instrumentsDataGridView.KeyDown += new
System.Windows.Form.KeyEventHandler(instrumentsDataGridView_KeyDown);
this.supportsDataGridView.KeyDown += new
System.Windows.Form.KeyEventHandler(supportsDataGridView_KeyDown);
        }
```

Fig. 7 (a) User Control

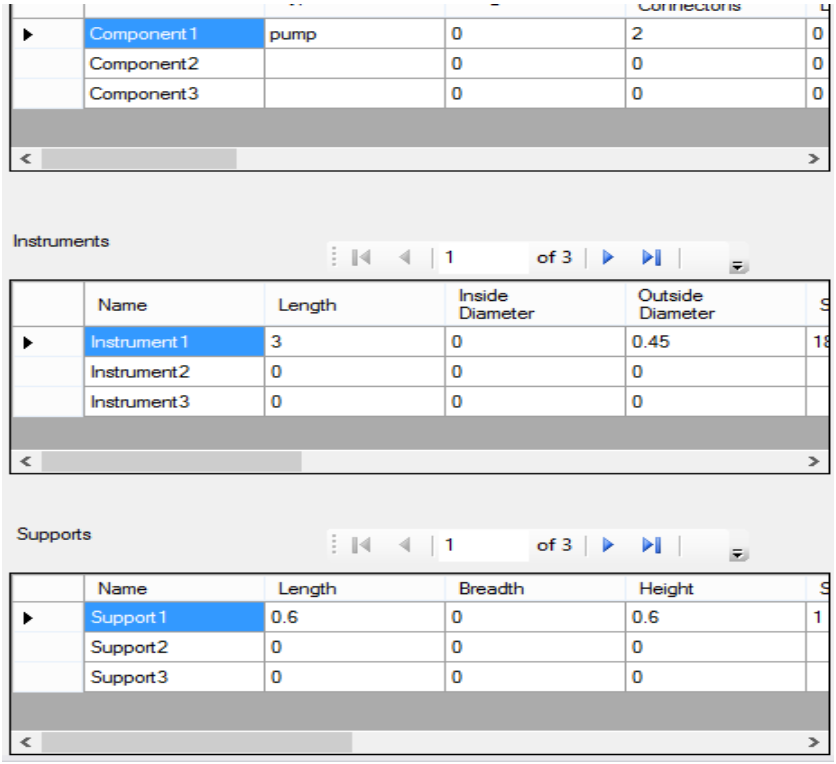

Fig. 7 (b) Modelling Editor

V. Conclusion and Future Work

It is important to note that DSML models are constructed using concepts that represent stakeholder's targets within the oil and gas pipeline application domain. Specification of a semantic model therefore means that the actual repository for the concepts of the language (i.e. its vocabulary) and their relations have to be organized to enhance communication during modelling. The modelling language internal mechanism can then enable it to follow the domain abstractions and semantics, allowing users to perceive themselves as working directly with domain concepts. With this capability of a well-defined semantic model, the artefacts are oriented during modeling at the level of stakeholder input through the defined concrete syntax editor to a solution that gives the result of the particular design intent. In this way, modeling through complicated platforms is carefully tackled. In the future, a high-level decision scheme can be specified to manipulate orientation standards within the language framework for easy processing of resource requests from the application model.

References

[1] Anvil International (2012), Pipe Fitters Handbook, University Park, IL 2012 United States.
[2] Alfred V. Aho, Ravi Sethi and Jeffrey D. Ullman, (2007), Compilers *Principles, Techniques, & Tools,* Pearson Education, New York.
[3] Braha D, Reich Y (2001) Topological structures for modelling engineering design processes. International conference on engineering design (ICED 01), Glasgow.
[4] *B.G. Technical* LTD - *B.G. Technical* Oil & Gas industry Port Harcourt, *Nigeria; www.bgtechnical.com/* Annual Reports 2009 to 2012.
[5] Bernhard Sch¨atz, fortiss GmbH, Guerickestr, From Solution to Problem Spaces: Formal Methods in the Context of Model-Based Development and Domain-Specific Languages, 35th IEEE Annual Computer Software and Applications Conference 2011.
[6] David A. Schmidt, (1997), Denotational Semantics: A methodology for language development *Department of Computing and Information Sciences, 234 Nichols Hall, Kansas State University, Manhattan, KS 66506. schmidt@cis.ksu.edu.*
[7] D. Méndez Fernández, D.C. Petriu, N. Rouquette, Ø. Haugen (Eds.): A Meta Model for Artefact-Orientation: Fundamentals and Lessons Learned in Requirements Engineering; MODELS 2010, Springer-Verlag Berlin Heidelberg 2010.
[8] Diomidis Spinellis, Notable design patterns for domain-specific languages, Department of Information and Communication Systems, University of the Aegean, GR-83 200 Karlovasi, Greece, The Journal of Systems and Software 56 (2001) 91±99.
[9] E. A. Lee and H. Zheng, "Operational semantics of hybrid systems," in Proceedings of Hybrid Systems: Computation and Control (HSCC), vol. LNCS 3414. Springer, 2005, pp. 25–53, invited Paper.
[10] Eric Evans (2003), Domain-Driven Design: Tackling Complexity in the Heart of Software Addison Wesley, USA.
[11] F. Zezula and C. Durden, (2000), Piping Joints Handbook, Piping & Pressure Systems Consultant, UTG, Sunbury.
[12] F. Klar, A. K¨onigs, and A. Sch ¨ urr, "Model Transformation in the Large," in Proceedings of the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering, ser. N. York: ACM Press, 2007, pp. 285–294.
[13] JieHu., Hubei Univ., Wuhan, China ; Mengchi Liu ; Junchi Zhang A Semantic Model For Academic Social Network Analysis Published in: Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference Beijing.
[14] Neelakantan R. Krishnaswami Nick Benton, A Semantic Model for Graphical User Interfaces ICFP '11 2011 ACM Tokyo.
[15] MattijsGhijsen, Jeroen van der Ham, Paola Grosso, CosminDumitru, Hao Zhu, Zhiming Zhao and Cees de Laat, A Semantic-Web Approach for Modeling Computing Infrastructures To appear in Computers and Electrical Engineering, 2013Universiteit van Amsterdam System and Network Engineering.SNE technical report SNE-UVA-2013-01http://sne.science.uva.nl/.