Domain Specific Modeling Language for Object Localization in Marine Observatories

Charbel Geryes Aoun*‡, Iyas Alloush*,
Yvon kermarrec*, Oussama Kassem Zein*
*Université européenne de Bretagne
*Telecom Bretagne, Institut Mines-Telecom
*UMR CNRS 6285 Lab-STICC
Bretagne, France
Email: charbel.aoun@telecom-bretagne.eu
iyas.alloush@telecom-bretagne.eu
yvon.kermarrec@telecom-bretagne.eu
oussama.zein@telecom-bretagne.eu

Joel Champeau‡
‡Ecole Nationale Supérieure de Techniques
‡Avancées Bretagne
Bretagne, France
Email: joel.champeau@ensta-bretagne.fr
charbel.aoun@ensta-bretagne.fr

***Abstract*—Marine observatories (MO) based on sensor networks provide a continuous observation of the ocean. The logical and physical components that are used in these observatories provide data exchanged environment between different devices (Smart Sensor, Data Fusion). These components provide new functionalities or services due to the stable running of this network. In this paper, we present our approach in extending the modeling languages to include new domain-specific concepts and constraints. Thus, we propose a meta-model that is used to generate a new design tool (ArchiMO). We illustrate our proposal with an example from the MO domain. Additionally, we generate the corresponding simulation code using our self-developed domain-specific model compiler. Our approach helps to reduce the complexity and time of the design activity. It provides a way to share the different viewpoints of the designers in the domain of MO.**

Keywords-Underwater Object Localization; Marine Observatories.

I. INTRODUCTION

Sensor network is a group of specialized sensors with a communications infrastructure designed to monitor and record terms at various locations. MO (Based on Sensor Networks) provide new opportunities to sea surveys, such as a continuous observation of the sea [1]. Our research scope is in the first phase of a MO project: Marine e-Data Observatory Network (MeDON) [2]. MeDON contains different elements (Hydrophones, Fusion Servers, Object Localization Algorithms), and different communication protocols (e.g., REST, SOAP) [2][3]. The implementation of this information system is considered as a complex distributed system [3]. We distinguish two sources of complexity: the complexity of the system, and the design. The complexity of the system under study [2][3] is related to: (1) the architecture of the system (Distributed) which contains different elements from different sub systems (Underwater Sensor Network and the rest of the information system); (2) the interactions between the different elements of the information system and the core network that relies on standard protocols and transactions; (3) the large number of sensors (Hydrophones) and servers existing on the networks.

Our scope in MeDON project is in the design of the Smart Sensor Network and the system to localize the underwater objects. Designing complex distributed systems consumes considerable time. According to [2][3], the complexity of the design is a result of: (1) the different domains of experience (Business Process Modeling, Information System Modeling, and the Underlying Infrastructure Modeling) that are required from the designer(s) to be able to model and describe such system; (2) distributed Software Structure of MeDON Information System (see Figure. 1) since each component (e.g., Data Fusion Server, Smart Sensor, etc.) is responsible to perform set of specific tasks.

Our global objective is to help the designers of MO to reduce: the complexity of the design and the time of the design activity. The deployment of set of sensors (Sensor Network) is an costly operation due to: the necessary equipments such as specific boats, marine cables, Sensors (Hydrophones), Data Fusion Servers, and experts in diving, etc. Additionally, we cannot ignore that the deployment operation is risky and the placement of sensors and servers should be in the right position where an error in meters may cause larger bit-error rates in the communication channel (Cables). Thus, an integration between the information system (Sensors, Servers) and the communication system (e.g., IMS) [4] is needed.

The large number of sensors that are communicating with set of fusion servers results in more complex design [5]. We consider that the time to obtain useful results from the MeDON system is the resultant of: the time of the operations of deployment and the design time. Thus, our research question is: how to improve the time of the design phase and reduce the complexity of the deployment and maintenance phase?

Our objective is to provide a design tool to the designers of MO that helps them to model their design taking into consideration reducing the time of development process, and managing the complexity.

In this paper, we propose a modeling design tool (ArchiMO) that helps to manage the complexity and prevents design modeling errors during the design time. This tool provides the designer a set of reusable graphical elements and concepts that respect ArchiMate [6] and the MO concepts. Our approach is based on the concept of domain specific modeling languages (DSMLs), which relies on Model Driven Engineers (MDE) fundamentals [7]. In order to model MO systems, we choose ArchiMate modeling language as it relies on Enterprise Architecture (EA) framework [8][9] that allows describing a

 ISBN: 978-1-61208-374-2

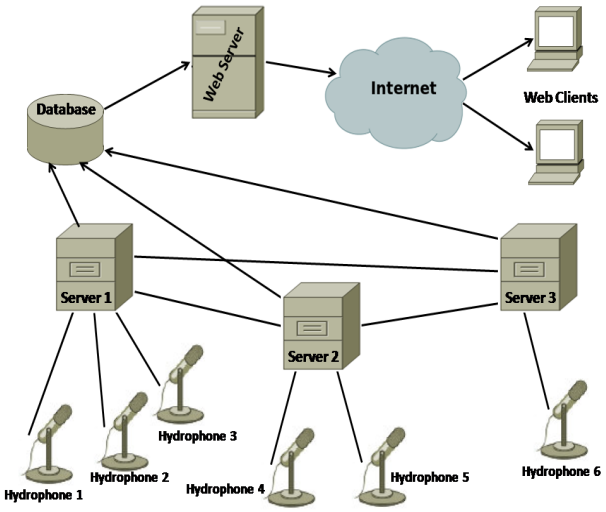

Fig. 1. Structure of MeDON- An Example: N=6, Y=3

wide range of domains [10]. We use meta-models to generate the tools that belong to different development activities using Eclipse Modeling Framework (EMF) [11].

ArchiMate is proper to model systems from the IT domain [6]. Our proposal extends the ArchiMate meta-model (Abstract and Concrete Syntax) to add new concepts and constraints of MO to ArchiMate. We add specific constraints to the grammar of the design tool according to the meta-model proposal. On one hand, a main feature of (EA) frameworks is sharing the multiple viewpoints [10]. This reduces complexity of one view to a manageable size. EA frameworks introduces interoperability issues between views and their dedicated software [10]. On the other hand, our proposed DSML is extensible, where the developers may extend it and add new concepts and standards according to the progress and needs in MO domain.

Linking our MO meta-model to the IP Multimedia Subsystem (IMS) one (proposed previously in [12] helps to integrate the different smart sensors of the sensor network to the rest of the information system through the core network [4]. We apply our design model to a model compiler to generate simulation code that runs directly in NS-3 network simulator [13].

The paper content is organized as follow: in Section II, we present the related work that is connected to the design tools. Section III presents MO project. In section IV, we present MDE fundamentals, DSMLs, ArchiMate, and our proposal meta-model for the MO/MeDON. Section V explains the abstract syntax, concrete syntax and semantics of the proposed DSML. In Section VI, we present the generated design tool and the simulation approach. In section VII, we conclude and discuss our future work.

II. Related Work

In this section, we present the related work in connection with the design tools.

In relation with the concept of Architectural Description Languages (ADLs) [14] and their design tools; we are interested in the following concerns that we shall specify and analyze in this section: (C1) preventing errors during design by invoking grammar or syntax of language; (C2) multiple viewpoints that are represented in the architectural description [15] since a viewpoint is a work product establishing the conventions for the construction, interpretation and use of architecture views to frame specific system concerns; (C3) extensibility of design tool; (C4) heterogeneity of components and communications; (C5) testing/execution platform.

According to the preventing errors concern, the design tool prevents errors during design activity that may be made by the designer, rather than correct them after the fact. This error prevention is available in [16][17][18]. Like in our approach, it's avoided by invoking the abstract syntax (Our Proposed Meta-Model) where we have defined and added our specific constraints and relations.

Concerning the multiple viewpoints concern, the design tool provides different viewpoints for the designers according to their specialties and domains of experience. In [16][17][18], the design tool provides only one viewpoint in order to fit software development tasks. This design tool does not provide the ability to share the design between different designers. Our approach considers this issue thanks to the different layers of EA standard that separates between perspectives.

Regarding the extensibility concern, the extension of a meta-model allows the extension of a design tool by adding new concepts and constraints to it [16][17]. It's realized in our approach by extending the ArchiMate meta-model by new elements and constraints, then generating a new design tool that contains the concrete syntax inside the palettes. These palettes contains the new added components like in [4][10].

Concerning the heterogeneity concern, the existence of different components and communications that are related to different contexts and activities. We are facing this heterogeneity in the software components and models in [16][17][18]. In our approach, we are facing this heterogeneity (e.g., Smart Sensor different than Data Fusion).

According to the execution test platform concern, the designer in [16][17][18] is not able to test and verify his models or instances on an executable platform (e.g., IMS). Relying on [4] in our approach, we are able to test our proposed model (see section VI). For example, IMS can be used to exchange messages between terminals (e.g. Smart Sensors, and Fusion Servers).

III. Marine Observatories

Underwater Sensor Networks that aims to environmental data acquisition will play an essential role in the development of future large data acquisition systems [19]. They allow the data to be exchanged and treated between the different devices (Servers, Sensors). On all these devices, we can have software components to treat and store the data. An example about MO is the project Marine e-Data Observatory Network (MeDON).

In this context, the designer should be able to include N acoustic sensors that are connected to the Y fusion servers as shown in (see Figure. 1). These servers treats the acoustic data

acquired by the hydrophones then diffuse them on the network. Servers store their data on the same database. The Database server provides the treated data to the web server where the configuration of a web application is done. Thus, the web server diffuses the information detected by the hydrophones such the voice of the dolphin to the web clients through a graphical interface.

IV. Model Driven Engineering (MDE) and Domain Specific Modeling Languages (DSML)

MDE [15] is "a software development method which focuses on creating and exploiting domain models. It allows the exploitation of models to simulate, estimate, understand, communicate, and produce code". MDE helps to manage complexity thanks to the modeling concept and model transformations. Modeling helps to describe the design in a high abstract way and model transformation helps to have a generated design tool.

A meta model defines by itself a language for describing a Specific Domain of interest [7]. In our approach, modeling tools follow the constraints and represent the concepts that are defined in the meta-model. It permits to instantiate large number of models that conform to it like in programming languages [20]; numerous of programs can be implemented relying on a specific programming language (e.g., C, C++, Java, etc.).

Eclipse IDE provides a powerful environment that relies on EMF which facilitates the modeling/meta-modeling activities, it supports many model transformation languages as well. Model transformations help us to generate design tools and simulation programs directly and automatically considering meta-models and model instances. Every model transformation depends on a set of rules that describe and control the transformation process. The transformation rules may map models that conform to different meta-models (on the same abstraction level), such as ATL [21], or map between different domains using one meta-model for the source model to generate texts/codes (e.g., XPAND [22]).

In our case (see Figure. 2), the input model represents the design of highly abstract level, and the meta-model is the extended ArchiMate meta-model which represents the abstract syntax [15][12]. Our code generation is an automated process that links directly the design model to the simulation scripts [13]. Thus, it helps to reduce the time of the implementations for large simulation programs, and it minimizes the implementation errors.

A. Domain-Specific Modeling Languages

Domain-Specific Modeling Languages (DSMLs) [23] enable designers from different domains and backgrounds to participate in software development tasks and to specify their own needs using domain concepts. A DSML [24] is comprised of three components: abstract syntax, concrete syntax, and semantics. The abstract syntax defines modeling concepts and their relationships. There are several kinds of concrete syntaxes: visual, XML-based, textual, etc [25]. The concrete

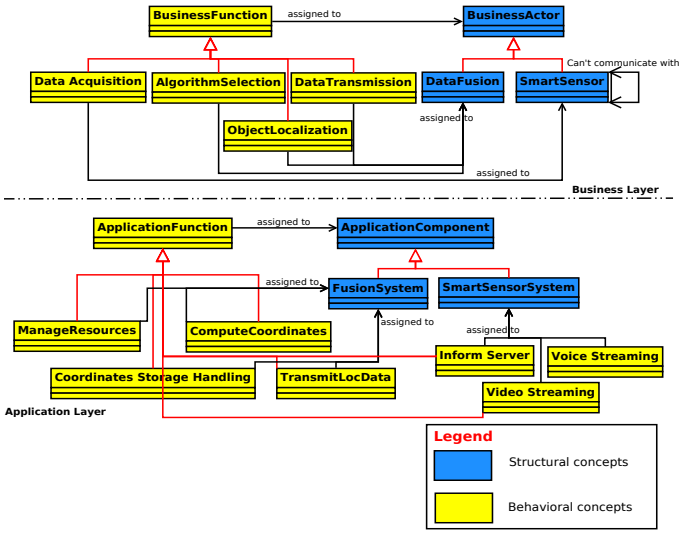

Fig. 2. Extending business and application layers of ArchiMate: proposal of MO Meta-Model

syntax is associated with a set of rules which defines the representation of the abstract syntax. Semantics describe the meaning of a model and are related to the abstract syntax. They are well-formed rules for the model and are used to constrain the concrete syntax [24].

Historically, data fusion methods were developed primarily for military applications (e.g., radars tracking a variable object) since fused data from multiple sensors provide several advantages over data from a single sensor [5]. We resume, such methodology as combining set of observations would result in an improved estimate of the target position. Concepts such information fusion and sensors networks have perforated the research and specially the military research. We distinguish different architecture for data fusion as follows [5][26]: (1) centralized fusion; (2) hierarchical fusion without feedback; (3) hierarchical fusion with feedback; (4) distributed fusion. According to our context, we have selected the most complex architecture (distributed) to model it, then simulate it in section VI. During the design activity, set of constraints and restrictions should be respected by the designer in order to model such architecture [5]. We will present them in the contribution section.

In general, errors caught during the design cycle are much less time consuming to identify and correct than those found during testing. In order to avoid errors in the design activity, we have implemented constraints that are defined in the abstract syntax of the language (Meta-Model) (see Figure. 2). The concrete syntax that is associated with these added constraints can be implemented in the design tool such as 'ArchiMO' tool in our context. This tool is generated relying on Eclipse-EMF (Tool Generation Concept thanks to Model Transformations).

Modeling languages are used to describe a system with high level of abstraction (e.g., UML 2.0) [25]. For MeDON/MO, and in relation with our objectives, we describe distributed sys-

 ISBN: 978-1-61208-374-2

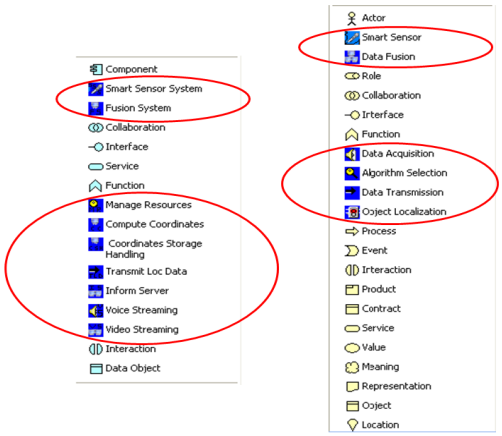

Fig. 3. Business and Application Layers (Palettes)

tems. UML is not enough to cover our needs, as it has only one layer that contains all of the concepts of the design, and these concepts are too general [27]. Thus, we selected ArchiMate modeling language that meets UML in some concepts, but it can describe the systems from IT domain and share multiple viewpoints during the design as it relies on TOGAF framework [15].

ArchiMate relies on Enterprise Architecture (EA) framework [9][15]. It decomposes the system design into three layers: business, application, and technology. In our approach, we present these layers in the following way:

1) Business layer: specifies the end-user functions and actors. It describes the service activities as perceived by the end-user, and the flow between them;
2) Application layer: specifies the functions and software components of the service. It describes the capability of the system under study, and the way of performing its tasks;
3) Technology layer: specifies the functions, topology, hardware elements, and signaling protocols of the underlying platform. It describes the execution platform that offers functions to be used by the functions of the application layer.

V. Contribution

In general, a meta-model of DSL represents the concepts/operations and constraints that belong to the domain specificities (MO in our case). In this section, we present our contribution of a new meta-model (Abstract Syntax), concrete syntax, and design tool. We extend the concepts of ArchiMate meta-model to represent the domain specifications of MO. The new meta-model enables us to generate and develop design tools that are coherent with Archi [28]. They contain additional concepts, elements, constraints and relations that are specific to the MO domain and for data fusion concepts [5].

Relying on the distributed fusion architecture (DFA) in [5], our meta-model (see Figure. 2), and according to [2], we distinguish the following constraints: for SmartSensor:*(1) communication between two Smart Sensor elements is not allowed; (2) communication between Smart Sensor and Data Fusion element is allowed; (3) Smart Sensor is only allowed to be related to the Data Acquisition function*. For DataFusion: *(1) communication between two Data Fusion elements is allowed; (2) Data Fusion is only allowed to be related to Algorithm Selection, Data Transmission and Object Localization functions.*

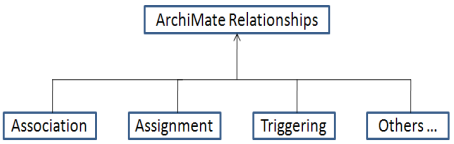

Fig. 4. ArchiMate RelationShips

Like in ArchiMate [9][15] our proposed meta-model is composed of two views: one for the business layer, and another for the application layer. Regarding the technology layer, we rely on a meta-model for IMS that provides an underlying platform in [4] to integrate the information system with the core network.

For each extended concept or element, a graphical view (belonging to the concrete syntax) should be defined [10]. Our proposed concrete syntax are shown in the palettes of the business (the red circles on the right of Figure. 3) and application (the red circles on the left of Figure. 3) layers. These palettes are coherent with MO specific concepts and relations from which the designer can select, drag and drop the desired ones.

ArchiMate contains different types of relationships such association, assignment, etc (see Figure. 4). We have specialized the definition of the relationships regarding the new added concepts. In our context, we have defined the association relationship for the smart sensor according to the constraints of DFA (e.g. smart sensor could be only associated to the data fusion). Furthermore, we have defined the assignment relationship for the smart sensor according to the constraint of MO [2] (e.g., Smart Sensor could be only assigned to the Data Acquisition).

For each ArchiMate element, we can define the relationship type that are allowed with this related element. The encoding ArchiMate relationship is based on an enumeration for all the possible types. The key values are for example 'o' for *association* and 'i' for the *assignment* relationship. For the business and application layers, we have implemented the keys relative to the selected relationships and mainly the associated constraints in java code regarding to our proposed extended meta-model (see Figure. 2). This implementation is the grammar of the new proposed DSML.

In order to have a graphical view for the added constraints and elements, we have generated the design tool ArchiMO relying on eclipse EMF.

This design tool helps the designer to model the system in a highly abstract way by drag and drop the elements and

 ISBN: 978-1-61208-374-2

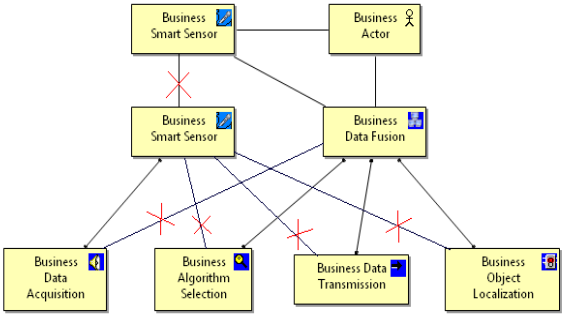

Fig. 5. Association and assignment relationships

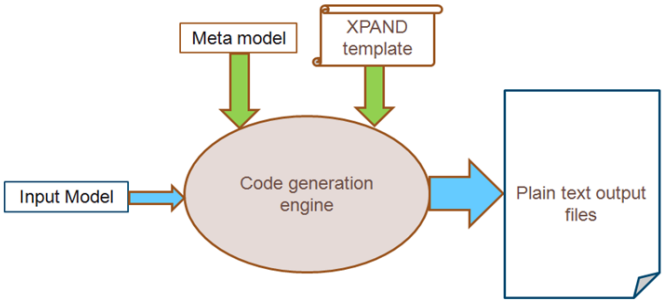

Fig. 6. The code generator workflow in XPAND language

relations from the palette. During the model edition, all the constraints specified for the MO extension are checked: (1) forbid the designer to associate two Smart Sensor elements together; (2) the designer is able to associate a Smart Sensor element to Data Fusion, Business Actor or other actors (see Figure. 5); (3) the assignment is only allowed from SmartSensor to the DataAcquisition function (see Figure. 5). Concerning the Data Fusion element: (1) the association between two Data Fusion elements is allowed; (2) the designer is able to associate Data Fusion element to Smart Sensor element (see Figure. 5); (3) the designer is able only to assign the Data Fusion to the Algorithm Selection, Data Transmission and Object Localization functions (see Figure. 5).

ArchiMO tool considers different domains of experience, each domain expert works in a specific layer (Business, Application or Technology) as the model created in section VI. Our contribution replies to the concerns that we have mentioned in II as it: (C1) prevent syntax and relation errors that can be made during the design activity; (C2) provides three layers according to each domain specificity; (C3) extends an open, standard, and classical design tool to have a specific one like ArchiMO; (C4) deploys different physical components (Sensors and Servers), and logical components such acquisition/localization algorithms.

VI. Object Localization Case Study

In order to validate our proposed tool, we use it to model the application of Object Localization using the different new elements that are proposed in the meta-model (see Figure. 2). Then we apply the design model to a model compiler (see Figure. 6) that we have developed to perform some error checks and generate automatically simulation code for NS-3. This simulation code runs in NS-3 tool that is a standard and classical simulator in the networking domain.

A. Design Model

We have modeled a system that localizes an underwater object using our generated design tool ArchiMO. In order to localize this object, sensors should be connected to data fusion servers. We have applied the distributed fusion architecture (DFA) [5] for this design.

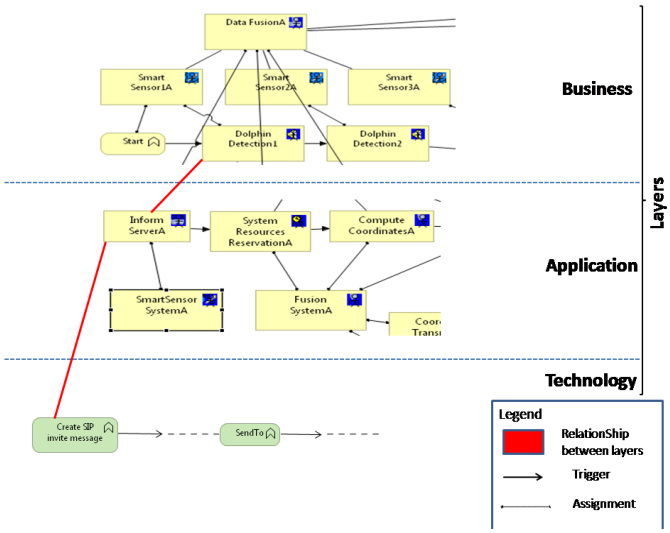

Fig. 7. Object Localization Underwater

The design model is composed of three views regarding to the layers of ArchiMate (see Figure. VI-A): Business, Application, and Technology. In Figure. VI-A, we present parts of the large model that is designed by ArchiMO. The model contains behavioral elements, in the business layer (see Figure. VI-A) shows the first activity of the smart sensor which is the dolphin detection1, etc. These activities are assigned to their proper smart sensors and these smart sensors are associated with the different data fusion servers and smart sensors that are required in the DFA [5]. Concerning the application layer, the behavioral elements are such the compute coordinates function that is triggered by the resources reservation function, and so on. Since ArchiMate allows the association between the layers, (see Figure. VI-A) shows this association. For example, the InformA Application Function aims to inform the fusion server A by the detection of an object through the smart sensor1A. Regarding the technology layer, a large series of functions are associated in it (e.g., *sendto*) to execute this application function. The *sendto* function forwards/sends a message of type SIP or Diameter from one node to another.

 ISBN: 978-1-61208-374-2

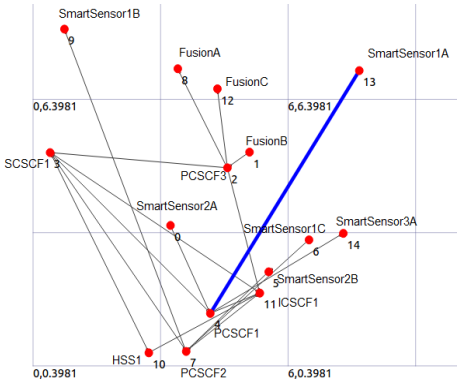

Fig. 8. Snapshot from the animation through NetAnim tool after running NS-3 simulation

B. Compilation and Simulation

The design tool ArchiMO generates an XMI file to represent the graphical design. This helps to conduct the design model to other tools. We use the XMI file as an input to our self-developed domain-specific model compiler to generate the simulation code (see Figure. 6). This hides complexity of constructing simulation programs from the designer and saves considerable time of the development process. The code generator needs both the meta-model including the abstract syntax of DSML for MO, and the input model that is generated from the design tool.

The XPAND template in (Figure. 6) contains the mapping rules between the model elements and their representations in NS-3 [13].

We have run the generated code in NS-3 (version 3.13), and the results of compilation and running shows no errors. Traces and logs (e.g., PCAP files) were generated to analyze the simulation outputs.

(Figure. 8) shows the architecture of the system design that is generated by NS-3 for the mentioned design model. NS-3 generated a hardware representations (Nodes, Interfaces, Wires) for the elements of the design model and the blue colored stream represents a message that is exchanged between two nodes in a fixed moment. This confirms that the behavioral elements were mapped as well.

We have used our approach in different application domains and network simulators (Video Conferencing System [12][13], and MO context). The common design concept between all cases is the underlying platform (IMS) that represents the Platform Specific Model (PSM) [25].

In other words, considering using one tool (e.g., NS-3), we could change the application domain relying on ArchiMate and our extensions (DSMLs) by fixing the underlying platform that is represented in the technology layer. This confirms that our proposed design tool (ArchiMO) creates models that follow the same meta-model and domain-specific concepts/constraints.

VII. Conclusion and Future Work

In this paper, we have presented a Domain Specific Modeling Language (DSML) for MO context. Our approach is based on extending the ArchiMate meta-model relying on MDE fundamentals. We have proposed a new design tool (ArchiMO) that is generated from the extended MO meta-model and respects the domain-specific concepts and constraints.

ArchiMO protects the designer from making design errors earlier than the other design activities and the code generation step. We are relying on a standard and open tool (Archi) and developing it by extending the modeling language and Java implementations. Another advantage is the extensibility of our proposed meta-model/tool. The developers may extend it and add new concepts and standards according to the progress in MO domain. ArchiMO provides the reusability of the added MO and Data Fusion concepts (e.g., Smart Sensor, Data Fusion, etc.) in different applications, activities, models or instances. ArchiMO reduces the time of the design activity as well, by having the specific elements and constraints in the palette of this tool. Additionally, we conserve the standard constraints in the abstract syntax (Meta-Model) of ArchiMate since the new added elements inherits concepts from standard ArchiMate elements.

On the other side, representing and meta-modeling the domain knowledge is itself a hard job that needs experience and high level of accuracy, especially when setting the grammar of the DSML according to the meta-model constraints.

As perspectives, we will extend our meta-model in order to satisfy and cover the most possible required operations, concepts and activities in the context of MO.

References

[1] O. Zein, J. Champeau, D. Kerjean, and Y. Auffret, “Smart sensor metamodel for deep sea observatory,” in *OCEANS 2009 - EUROPE*, May 2009, pp. 1–6.

[2] *MeDON - Acoustic Data. URL: http://www.medon.info/.*, Last visited 03-November-2014.

[3] J.-P. Schneider, J. Champeau, and D. Kerjean, “Domain-specific modelling applied to integration of smart sensors into an information system,” in *International Conference on Enterprise Information Systems (ICEIS 2011)*, Lille, France, Jun. 2011.

[4] V. Chiprianov, I. Alloush, Y. Kermarrec, and S. Rouvrais, “Telecommunications service creation: Towards extensions for enterprise architecture modeling languages,” in *6th Intl. Conf. on Software and Data Technologies (ICSOFT)*, vol. 1, Seville, Spain, 2011, pp. 23–29.

[5] M. E. Liggins, D. L.Hall, and J. Llinas, *Multisensor Data Fusion, Theory and Practice*, S. edition, Ed. Taylor & Francis Group, LLC, 2009.

[6] *The Open Group, ArchiMate 1.0 Specification. http://www.opengroup.org/subjectareas/enterprise/archimate*, Last visited 03-November-2014.

[7] J.-L. Pérez-Medina, S. Dupuy-Chessa, and A. Front, “A survey of model driven engineering tools for user interface design,” in *Proceedings of the 6th International Conference on Task Models and Diagrams for User Interface Design*, ser. TAMODIA’07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 84–97.

[8] O. Noran, “An analysis of the zachman framework for enterprise architecture from the {GERAM} perspective,” *Annual Reviews in Control*, vol. 27, no. 2, pp. 163 – 183, 2003.

[9] D. Quartel, W. Engelsmanb, H. Jonkersb, and M. van Sinderenc, “A goal-oriented requirements modelling language for enterprise architecture,” in *Enterprise Distributed Object Computing Conference, 2009. EDOC ’09. IEEE International*, University of Twente. IEEE, 2009, pp. 3 – 13.

[10] V. Chiprianov, Y. Kermarrec, and S. Rouvrais, “Extending enterprise architecture modeling languages: Application to telecommunications service creation,” in *The 27th Symposium On Applied Computing*. Trento: ACM, 2012, pp. 21–24.

[11] *Eclipse Modeling FrameWork. http://www.eclipse.org/modeling/emf/*, Last visited 03-November-2014.

 ISBN: 978-1-61208-374-2

[12] I. Alloush, V. Chiprianov, Y. Kermarrec, and S. Rouvrais, "Linking telecom service high-level abstract models to simulators based on model transformations: The IMS case study," in *Information and Communication Technologies (EUNICE 2012)*, ser. Lecture Notes in Computer Science, R. Szabó and A. Vidócs, Eds., vol. 7479. Springer Berlin Heidelberg, August 2012, pp. 100–111.

[13] I. Alloush, Y. Kermarrec, and S. Rouvrais, "A generalized model transformation approach to link design models to network simulators: Ns-3 case study," in *International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2013)*. SciTePress Digital Library, July 2013, pp. 337–344.

[14] N. Medvidovic and R. Taylor, "A classification and comparison framework for software architecture description languages," *1*, vol. 26, pp. 70–93, Jan 2000.

[15] V. Chiprianov, "Collaborative construction of telecommunications services. an enterprise architecture and model driven engineering method," Ph.D. dissertation, Telecom Bretagne, France, 2012.

[16] L. Touraille, M. K. Traoré, and D. R. C. Hill, "A model-driven software environment for modeling, simulation and analysis of complex systems," in *Proceedings of the 2011 Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium*, ser. TMS-DEVS '11, San Diego, CA, USA, 2011, pp. 229–237.

[17] K. Y. A. Achilleos and N. Georgalas, "Context modelling and a context-aware framework for pervasive service creation: A model-driven approach," *Pervasive and Mobile Computing*, vol. 6, no. 2, p. 281–296, 2010.

[18] J.-L. Bakker and R. Jain, "Next generation service creation using xml scripting languages," vol. 4, pp. 2001–2007, 2002.

[19] J. Sorribas, A. Barba, E. Trullols, J. Del Rio, A. Manuel, and M. de la Muela, "Marine sensor networks and ocean observatories. a policy based management approach," in *Computing in the Global Information Technology, 2008. ICCGI '08. The Third International Multi-Conference on*, July 2008, pp. 143–147.

[20] J. Bezivin, "In search of a basic principle for model driven engineering,," *Novatica Journal*, vol. vol. 2, p. pp. 21–24, 2004.

[21] *Atlas transformation language. http://www.eclipse.org/atl/*, Last visited 03-November-2014.

[22] *Eclispe Modeling. http://www.eclipse.org/modeling/*, Last visited 03-November-2014.

[23] M. M. T. Zekai Demirezen, Barrett R. Bryant, "Dsml design space analysis," in *UAB, Birmingham, AL 35294, USA*, 2011.

[24] H. Cho, J. Gray, and E. Syriani, "Creating visual domain-specific modeling languages from end-user demonstration," in *Modeling in Software Engineering (MISE), 2012 ICSE Workshop on*, June 2012, pp. 22–28.

[25] I. Kurtev, J. Bézivin, F. Jouault, and P. Valduriez, "Model-based DSL frameworks," in *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*, ser. OOPSLA '06. New York, NY, USA: ACM, 2006, pp. 602–616.

[26] I. Liggins, M.E., C.-Y. Chong, I. Kadar, M. Alford, V. Vannicola, and S. Thomopoulos, "Distributed fusion architectures and algorithms for target tracking," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 95–107, Jan 1997.

[27] I. Sommerville, *Sofware Engineering, Ninth Edition*, M. Horton, Ed. Pearson, 2011.

[28] *Archi tool. http://archi.cetis.ac.uk/developer/model-new-element.html*, Last visited 03-November-2014.