A Mapping Approach for Marine Observatory Relying on Enterprise Architecture

*†Charbel Geryes Aoun, *Iyas Alloush, *Yvon kermarrec, †Joel Champeau, ‡Oussama Kassem Zein
*Université européenne de Bretagne, Telecom Bretagne, Institut Mines-Telecom, UMR CNRS 6285 Lab-STICC
†Ecole Nationale Supérieure de Techniques Avancées, ENSTA Bretagne, Bretagne France
‡Lebanese University, Faculty of Sciences, Lebanon
Email: charbel.aoun@telecom-bretagne.eu, charbel.aoun@ensta-bretagne.fr
iyas.alloush@telecom-bretagne.eu, yvon.kermarrec@telecom-bretagne.eu
joel.champeau@ensta-bretagne.fr, oussama.zein@ul.edu.lb

***Abstract*—UnderWater Sensor Networks (UW-SNs) performs collaborative monitoring tasks over an underwater determined area. These tasks could be underwater or deep sea observatories. Acoustic sensors (Hydrophones) are responsible to acquire the data underwater then transfer it to components/devices. Marine Observatory (MO) is the scenario of data exchanged between the different components/devices of the Underwater Acoustic Sensor Networks (UW-ASNs). Hence, the MO infrastructure could be based on an (UW-ASNs). This observation should take into consideration the environmental constraints since it may require specific tools, materials and devices (marines cables, specific servers and routers, etc.). The logical and physical components that are used in these observatories supply interchange procedures between the various devices of the environment (Smart Sensors, Data Fusion Servers). These components provide new services due to the long period running of the network. In this paper, we present our extended Meta-Model that is used to generate a new design tool (ArchiMO). Thus, we propose a mapping approach between the layers of the enterprise architecture standard. This approach throws instantly domain-specific concepts and constraints in layers on run-time design activity according to a satisfaction of a certain criteria/constraints. We illustrate our proposal with an underwater object localization example from the MO domain. Additionally, we generate the corresponding simulation code for a standard network simulator using our self-developed domain-specific model compiler. Our approach helps to manage the complexity, and to reduce the time of the entire development life cycle of an MO information system. It supplies in the MO context, a way to share the different viewpoints of the designers.**

***Keywords*—Smart Sensors, Data Fusion, ArchiMate, Enterprise Architecture, Meta-Model, Model Driven Engineers, Underwater Object Localization, Marine Observatory, Models, Simulation, Domain Specific Modeling Language.**

I. Introduction

Marine Observatory (MO) applications can include Underwater Acoustic Sensor Networks (UW-ASNs) that consist of a variable number of sensors that are deployed to gather scientific data in collaborative monitoring missions [1]. These types of applications monitor environment data of different fields and relays information in real-time to a dedicated workstation (e.g. Wide Area Network). To achieve this objective, sensors self-organize in an autonomous network (e.g. UW-ASNs) which can adapt to the characteristics of the ocean environment (e.g. set of hydrophones and marine cables). Sensor Networks (SN) consist of devices with varying levels of computational and communication capabilities [2]. In addition, SN in [3] operate as a distributed information system with the capability to answer queries to the user regarding some service requirements. Hence, a sensor network is considered to be a complex system since it aggregates all these features.

Our research scope is applied to the first phase of a MO project: Marine e-Data Observatory Network (MeDON) [4]. It's similar to any MO project, MeDON provides new opportunities to sea surveys, such as a continuous observation of the sea [1]. In order to assure this observation, the infrastructure of these types of projects is based on localization systems [5]. MeDON contains different elements (Hydrophones, Fusion Servers, Object Localization Algorithms), and different communication protocols (e.g. REST, SOAP and proprietary ones) [4][6]. Thus, an integration between MeDON and communication system, like the IP Multimedia Subsystem (IMS) is needed since it's useful to ease the integration of heterogeneous components with the internet [7].

It's obvious that the implementation of MO information system is considered as a complex distributed system [6]. We distinguish three sources of complexity: the complexity of the system itself, the specification and design activities, and the deployment. The complexity of the system under study [4][6] is related to: (1) the architecture of the distributed system which contains different elements from different sub systems (UW-ASNs 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 software and hardware components that are existing on the network (e.g. Smart-Sensors/Hydrophones, Logical Algorithmic Functions, Data Fusion Servers, Physical and Logical connections between the different components, etc.).

According to [4][6], the complexity of the design is also emphasized of: (1) the different domains of experience (Busi-

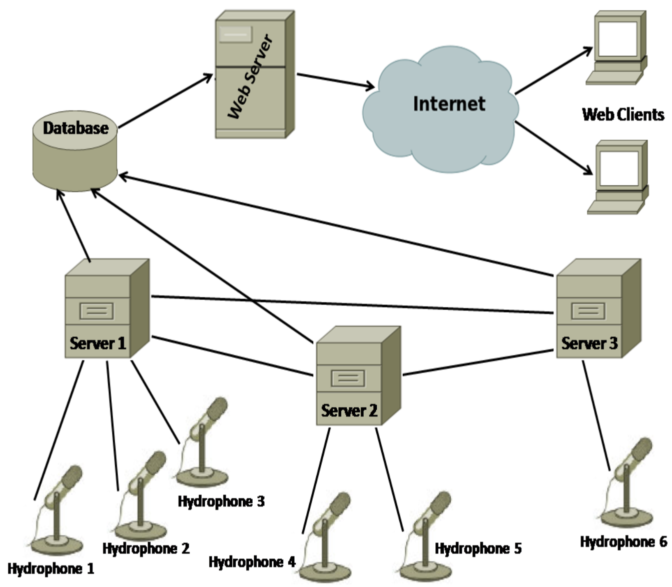

Fig. 1. Structure of MeDON- An Example: N=6, Y=3

ness 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 dedicated tasks.

Due to these aspects, designing complex distributed systems consumes considerable time due to the element heterogeneity (e.g. physical, logical), the high element number and these elements are related together from the business scenario viewpoint, to the physical communication viewpoint. In order to relate two concepts in the business viewpoint (e.g. Smart Sensor, Fusion Server), a large number of concepts and relations is needed in the application viewpoint (e.g. 7 concepts and 9 relationships), and in the technology/communication viewpoint the number of needed concepts and relations increase more and more. Also, a lot of difficulties and obstacles face the designers during their cooperation tasks in order to have an overall model. In order to have this model, the designers at different layers should work and react between each other like a collaborative team.

Our research question is: ***how to hide the complexity of the shared operation between designers and reduce the time of the design phase?*** In this context, our objective is to provide a design tool to the designers of MO that helps them to model their design taking into consideration: (1) managing the complexity that will face the designers during their cooperative tasks; (2) reducing the time needed to design in different domains of experience a large number of components and relations.

In this paper, we propose a modeling design tool (ArchiMO) that helps manage the complexity and provide automatic models with the large specific number of concepts and relationships during the design time. This tooling provides the designer a graphical modeling language that respect ArchiMate concepts [8] 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 [9]. In order to model MO systems, we choose ArchiMate modeling language as it relies on Enterprise Architecture (EA) framework [10][11] that allows describing a wide range of domains [12]. We use Meta-Models to generate the tools that are specified to different development activities using Eclipse Modeling Framework (EMF) [13].

ArchiMate is defined intended to model systems from the IT domain [8]. Our proposal extends the ArchiMate Meta-Model to add new specific constraint for MO into ArchiMate. On one hand, a main feature of EA frameworks is sharing the multiple viewpoints [12]. This reduces the complexity of the three views (Business, Application and Technology layers) to a manageable set of concepts. EA frameworks introduce interoperability issues between views and their dedicated software [12]. On the other hand, our proposed constraints is extensible, where the developers may extend it and implement new features and criteria according to the needs in MO domain.

To integrate the different smart sensors of the sensor network to the rest of the information system through the core network [7], we link our MO Meta-Model to the IP Multimedia Subsystem (IMS) one (proposed previously in [14]). Next, we apply our design model to a model compiler to generate simulation code that is linked to the NS-3 network simulator [15]. The resulted simulation helps to obtain an early validation of the design model before any prototype or deployment.

The paper content is organized as follows: section II presents the related works that are connected to our design tools. In Section III, we present the context, MO domain. Section IV presents MDE fundamentals, DSMLs and ArchiMate. Our contribution which extends the ArchiMate Meta-Model with the MO/MeDON concepts is described in the section V. In Section VI, we present the generated design tool and the simulation approach. And finally in the section VII, we conclude and discuss our future work.

II. Related Work

In this section, we want to detail the requirements/needs of the designers during the design activity. In order to have specific models/instances, the designers recommend special modeling languages during their use of the design tools. Hence, we present the related work in connection with the design tools.

In relation with the concept of Architectural Description Languages (ADLs) [16] and their design tools; we are interested in the following concerns/requirements that we shall specify and analyze in this section: (C1) getting automatic models in specific viewpoints by extending the constraints of the syntax language; (C2) collaborative team support is needed, so that the designers and developers can communicate;

(C3) preventing architectural errors during design by invoking grammar or syntax of language; (C4) multiple viewpoints that are represented in the architectural description [17] since a viewpoint is a basic product establishing the conventions for the construction, interpretation and use of architecture views to frame specific system concerns; (C5) extensibility of design tool and having specific views to tasks and roles; (C6) heterogeneity of components and communications; (C7) testing/execution platform.

Regarding getting automatic models concern, these models are thrown from a viewpoint by the designer to other viewpoints and designers. The design tool in [18][19][20] doesn't provide on run-time of the design activity, the ability to throw the models directly and quickly from the business layer to the application or technology layers. This ability is interested since some domain experts (e.g. designers) can get their specific concepts and relationships easily and automatically. Our approach takes this issue into consideration, thanks to the different layers of EA, that separates between perspectives since each layer is dedicated for a domain expert. Relying on this separation, we add the necessary constraints to the grammar of the language in order to invoke on run-time of the design activity, the related concepts and relationships for each specified layer.

Regarding the collaborative team support concern, a cooperation between the different domain experts (designers) to have one overall integrated model is highly recommended in order to exchange their knowledge according to their domains of experience. The support of this cooperation is not faced in [18][19][20]. Our approach considers this issue by supporting the communication between the different designers when invoking on run-time of the design activity for the specific concepts and relationships of each domain experts/designers.

According to the preventing architectural errors concern, the design tool prevents errors during design activity that may be made by the designer, rather than correct them after deploy. This error prevention is available in [18][19][20]. Our approach also prevents this error 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 [18][19][20], the design tool provides only one viewpoint in order to fit only software development tasks. This design tool does not provide the ability to share the design between different designers. This ability is essential since different domain experts can share the same design. Our approach considers this issue thanks to the different layers of EA standard that separates between perspectives since each layer is dedicated for a domain expert.

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 [18][19]. 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 [7][12].

Concerning the heterogeneity concern, the existence of components and communications that are related to different contexts and activities. This heterogeneity in the software components and models is realized in [18][19][20]. In our approach, we are facing this heterogeneity (e.g. Smart Sensor different than Data Fusion) but we are able to solve this issue by relating the different existing devices together according to a specific scenario.

According to the execution test platform concern, the designer in [18][19][20] is not able to test and verify his models or instances on an executable platform (e.g. IMS). Relying on [7] in our approach, we are able to test our proposed model (see section VI).

III. Context

An essential role is played by the Underwater Sensor Networks that aim to environmental data Acquisition, in the development of future large data acquisition systems [21]. They allow the data to be exchanged and processed between the different devices (Servers, Sensors). On all these devices, we can have software components to process and store the data. An example about MO is the project Marine e-Data Observatory Network (MeDON). MeDON is a seafloor observatory information system made of several sensors and a computing system aiming at detecting and localizing marine mammals through Passive Acoustic Monitoring [22]. The MeDON project [4] is an interesting example for the use of multiple modeling formalism.

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). The acoustic data acquired by the hydrophones, are analyzed and processed by these servers, then diffused on the network. Servers store their data on the same database. Then the processed and filtered data are provided by the Database server to the web server where the configuration of a web application is done. Thus, the web server broadcasts the information detected by the hydrophones such the dolphin acoustic sounds to the web clients through a graphical interface.

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

MDE [17] 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. Thanks to the modeling concept and model transformations, MDE helps to manage complexity.

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 a language for describing a Specific Domain of interest [9]. In our approach, modeling tools contains the specific constraints and represents the concepts,

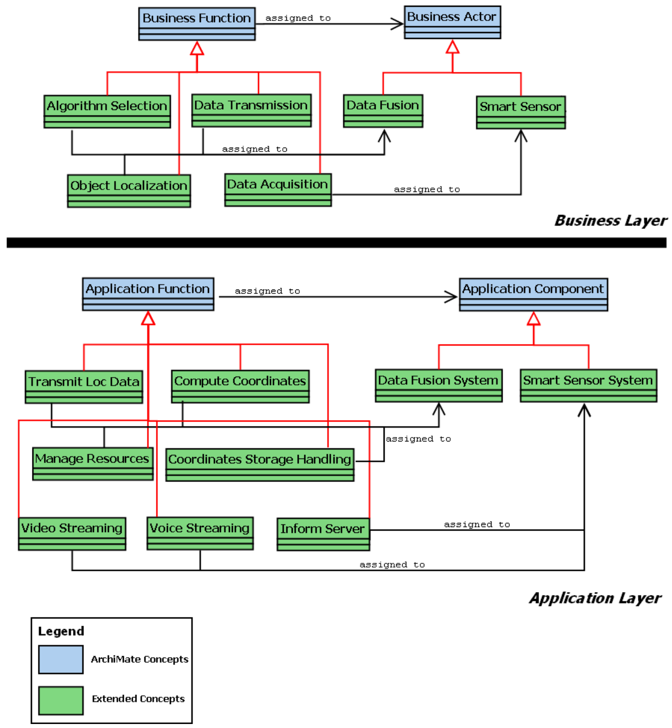

Fig. 2. Extending business and application layers of ArchiMate: proposal of MO Meta-Model

elements and relations that are defined in the Meta-Model. It makes it possible to instantiate large number of models that conform to it like in programming languages [23]; 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, such as ATL [24], or map between different domains using one Meta-Model for the source model to generate texts/codes (e.g. XPAND [25]). We conclude that the main benefit or objective of the model transformation concept is the ability of changing from technological space to another technological space (e.g. from xml extension to java or c++).

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 [17][14]. Our code generation is an automated process that links directly the design model to the simulation scripts [15]. Thus, it helps to reduce the time of the implementations for large simulation programs, and it minimizes the implementation errors.

Domain-Specific Modeling Languages (DSMLs) [26] enable designers from different domains and backgrounds to participate in software development tasks and to specify their own needs in a specific context using domain concepts, objects, classes, elements and entities. A DSML [27] 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 [28]. The concrete 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 [27].

Data fusion is the process of combing information from a number of different sources to provide a robust and it's in any application where a large amount of data must be combined or fused to obtain interested information [29]. This process is the approach that provides the identification of the area of the moving object relying on minimum two sources. Concepts such information fusion and sensors networks have dominated the research and specially the military research.

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 [30]. A methodology as combining set of observations would result in an improved estimate of the target position by minimizing the error that may occur by calculating the position of a variable object. Relying on [31], a high position accuracy of mobile object is assured when the number of sources increases. We select the data fusion approach to be implemented in our proposed localization service. In order to select the proper architecture for the data fusion, we will elaborate below the different existing types of Data Fusion Architectures.

Mainly, there are four possible architectures: centralized, hierarchical without feedback, hierarchical with feedback, and fully distributed [30][29][32]. In a centralized architecture, there is a single fusion node, in the hierarchical architectures, the fusion nodes are arranged in a hierarchy with the higher level nodes processing results from the lower level nodes and possibly providing some feedback [30][29][32]. In a fully distributed architecture, there is no pre-determined superior/subordinate relationship, each node can communicate with any other node subject to connectivity constraints, and the communication can be asynchronous [30][29][32]. Basically, when sensors could communicate information directly to a central processor where it is combined, it's the simplest architecture which is typically as the centralized one [29]. According to [32], in many applications, distributed fusion is a better architecture than centralized relying on some advantages such as distribution of processing load, lower communication, bandwidth, local authority, higher reliability, etc [32]. According to our context, we have selected the most complex architecture (distributed) to model it, then simulate it as presented in section VI.

During the design activity, set of constraints and restrictions should be respected by the designer in order to model distributed system architecture like in [30]. We will present them in the next section.

In general, errors caught during the design cycle are much less time consuming to identify and correct than those found during the testing phase. 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 [28]. For MeDON/MO, and in relation with our objectives, we describe distributed systems. Thus, we selected ArchiMate modeling language that can describe the systems from IT domain and share multiple viewpoints during the design as it relies on TOGAF framework [17].

ArchiMate relies on Enterprise Architecture (EA) framework [11][17]. 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 our context, we based our work on ArchiMate which relies on Enterprise Architecture (EA) framework [11][17]. It presents 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 flows 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 communicating the different existing elements and relations in order to perform 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.

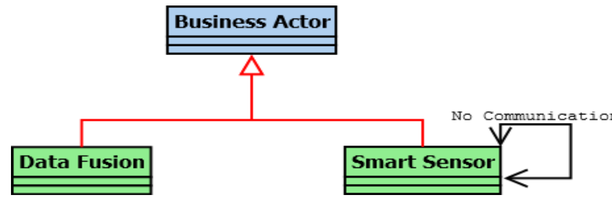

Fig. 3. Communication constraint between two Smart Sensors

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), new runtime mapping approach between the different layers of the EA, concrete syntax, design tool. The concepts of ArchiMate Meta-Model in [33] (see Figure. 2) are extended to represent the domain specifications of MO.

Our proposed Meta-Model in [33] is composed of two views: one for the business layer, and another for the application layer. Regarding to the technology layer, we rely on a Meta-Model for IMS underlying platform that we proposed before in [12]. We present the proposed Meta-Model as the following: (1) Business Layer (see Figure. 2): we have extended the business actor of ArchiMate by two new concepts: the Smart Sensor, and the Data Fusion (the green classes or concepts in the Business Layer in Figure. 2). We have extended also the business function of ArchiMate by four new concepts: the Algorithm Selection, Data Transmission, Object Localization, and the Data Acquisition (the green classes or concepts in the Business Layer in Figure. 2); (2) Application Layer (see Figure. 2): we have extended the application component by two elements: the Data Fusion System, and the Smart Sensor System (the green classes or concepts in the Application Layer in Figure. 2). We have extended also the application function of ArchiMate by seven new concepts: the Manage Resources, Coordinates Storage Handling, Compute Coordinates, Transmit Localization Data, Voice Streaming, Video Streaming, and Inform Server (the green classes or concepts in the Application Layer in Figure. 2).

Some specific communication constraints are added to the ArchiMate Meta-Model in [33], such as specific constraints for Smart Sensor in [34] (see Figure. 3). Also similar to a new relationship which is SmartSensor DataFusion Relation (SDR) in [35] that is used only to connect a Smart Sensor element to a Data Fusion element by giving proper values by the designers with respecting the existing criteria between Smart Sensor and Data Fusion concepts (see Figure. 4). This SDR constraint is similar to any ArchiMate Relationship. And to establish a relationship between two ArchiMate elements a certain criteria should be respected and satisfied (see Figure. 5).

An extension to the added communication constraint in [35] which is the SDR, is implemented in its abstract syntax. This extended constraint is responsible to call specific extended MO classes and relationships from the abstract syntax of the

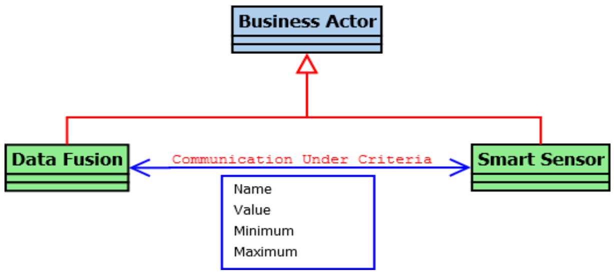

Fig. 4. Communication constraint between Smart Sensor and Data Fusion Server

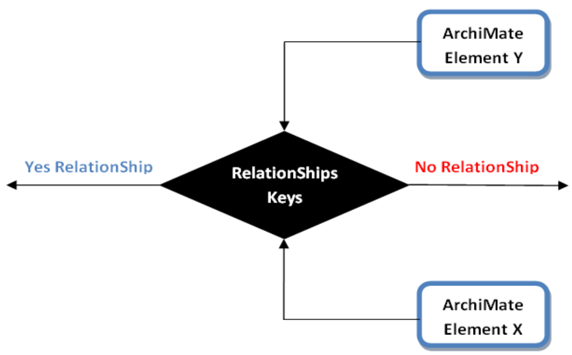

Fig. 5. Relationship between two ArchiMate Elements

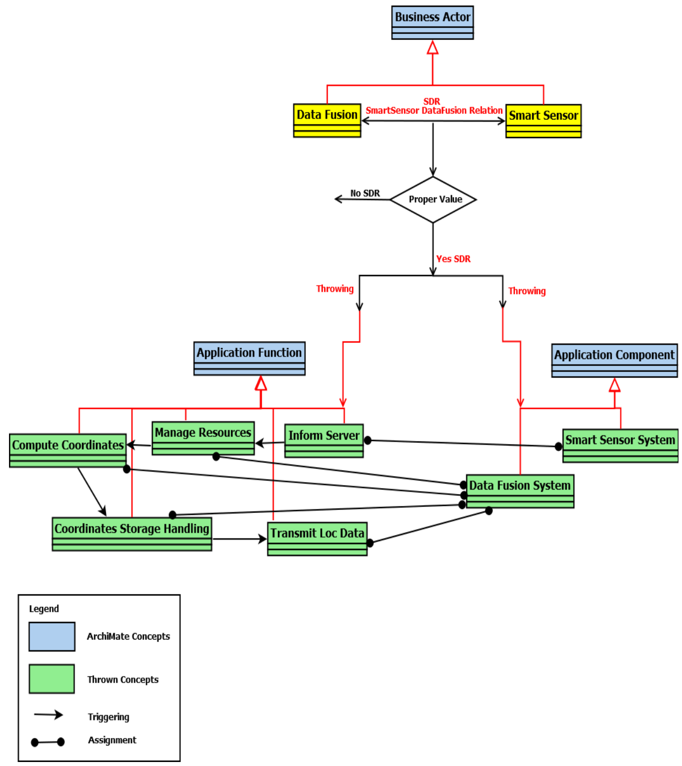

Fig. 6. Thrown MO Concepts and Relationships

application layer, then throw them in the viewpoint of the application layer. It can invoke and also throw the ArchiMate classes and relationships with the same scenario. This mix invoking between MO and ArchiMate concepts is elaborated in Figure. 6, since the triggering and the assignment relationships are ArchiMate elements and not extended MO elements like Smart Sensor System, Compute Coordinates, etc.

The extension of SDR constraint enables the designer of the business layer according to specific MO activities in the business layer, to throw specific extended MO and ArchiMate application layer concepts in the viewpoint of the application layer. Once the designer of the business layer finishes the design of MO activities, the designer of the application layer will get automatically his specific MO models/instances. For example, when the designer of the business layer connects two MO elements (e.g. SmartSensor and DataFusion), the necessary corresponding MO elements and relations are thrown in the application layer (see Figure. 6).

The new Meta-Model enables us to generate and develop design tools that are coherent with Archi [36]. They contain additional concepts, elements, constraints and relations that are specific to the MO domain and for data fusion concepts [30].

Relying on the distributed fusion architecture (DFA) in [30], our Meta-Model (see Figure. 2, Figure. 3 and Figure. 4), and according to [4], we distinguish the following constraints: for SDR: *(1) throwing corresponding required MO concepts in the application layer; (2) throwing corresponding required ArchiMate concepts in the application layer*. For Smart Sensor: *(1) communication between two Smart Sensor elements is not allowed; (2) communication between Smart Sensor and Data Fusion element is allowed; (3) communication between Smart Sensor and Data Fusion is allowed according to a specific constraint; (4) Smart Sensor is only allowed to be related to the Data Acquisition function*. For Data Fusion: *(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*.

Like in ArchiMate [11][17] 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 [7] to integrate the information system with the core network.

For each extended concept, element or Relationship, a graphical view (belonging to the concrete syntax) should be defined [12]. Our proposed concrete syntax in [34] are shown in the palette of the business layer (the right red circles in Figure. 7) and the application layer (the left red circles in Figure. 7). Also, we proposed concrete syntax in [35] and its shown in the palette of the Relationships (the red circle in Figure. 8). 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 as association, assignment, etc (see Figure. 9). We have extended the association relationship in [35] by a new one which is the SDR. The new extended Relationship is shown in red (see

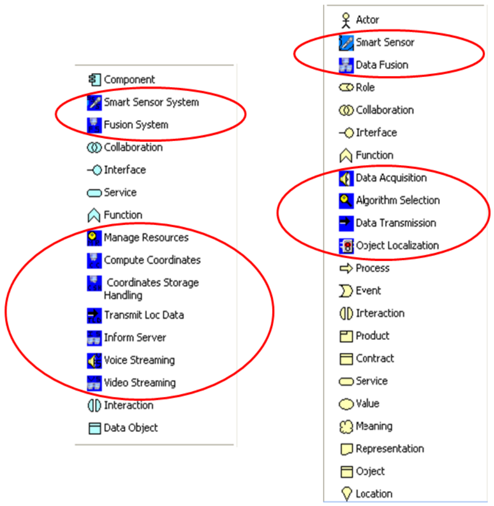

Fig. 7. Generated Business and Application Layers in Palette

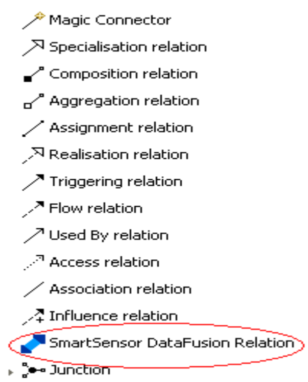

Fig. 8. New Relationship in Palette

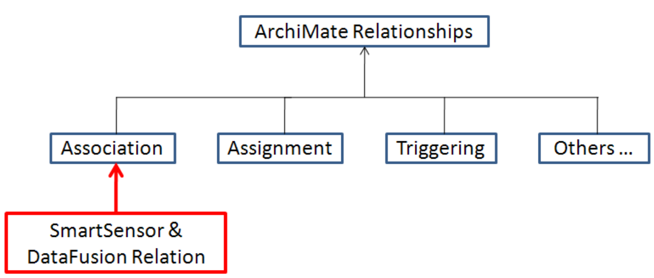

Fig. 9. ArchiMate RelationShips

Figure. 9).

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. We have added the key value 'x' for the new extended SDR. We have implemented this key in java code based on our proposed extended relationship (see Figure. 9). We have extended this java code by invoking the different necessary MO classes and ArchiMate relationships from the application layer that should be thrown once the SDR is well created by the designer of the business layer (see Figure. 6). This implementation is the grammar of the new proposed mapping approach.

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 relations from the palette. During the model edition, most of the specified constraints for the MO extension are checked and verified in [34]. The proposed SDR communication constraints are checked and verified in [35].

The extension of SDR constraints is checked and verified in this paper. Once the association of a Smart Sensor to a Data Fusion using the SDR is well established after entering a proper value by the designer of the business layer (see Figure. 10):

(1) required and necessary MO elements and relationships are thrown in the application layer (Inform Server, Manage Resources, Coordinates Storage Handling, Compute Coordinates, Transmit Localization Data, Smart Sensor System, Data Fusion System). In the application layer of the Figure. 10, Inform Server class is represented by Inform ServerA instance, Manage Resources class is represented by System Resources ReservationA instance, Compute Coordinates class is represented by Compute CoordinatesA instance, Coordinates Storage Handling class is represented by Storage CoordinatesA instance, Transmit Loc Data class is represented by Coordinates Transmission To B or Coordinates Transmission To C instance, Smart Sensor System class is represented by Smart Sensor SystemA instance, and Data Fusion System class is represented by Fusion SystemA instance;

(2) required ArchiMate elements and relationships are thrown in the application layer (triggering and Assignment are ArchiMate relationships). In the application layer of the Figure. 10, triggering relation is represented by the different existing relation between the following instances of MO classes: Inform ServerA, System Resources ReservationA, Compute CoordinatesA, Storage CoordinatesA, Coordinates Transmission To B or Coordinates Transmission To C. Assignment relation is represented by the relation between the Smart Sensor SystemA and Inform ServerA instances, and between the Fusion SystemA with the rest of existing instances.

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) provides a mapping approach between the different layers of EA to throw directly the needed components and relations to each domain expert or designer; (C2) facilitate the communication and interactions between the different concerned designers within the same model; (C3) prevent syntax and relation architectural errors that can be made during the design activity; (C4) provides three layers according to each domain specificity; (C6) extends an open,

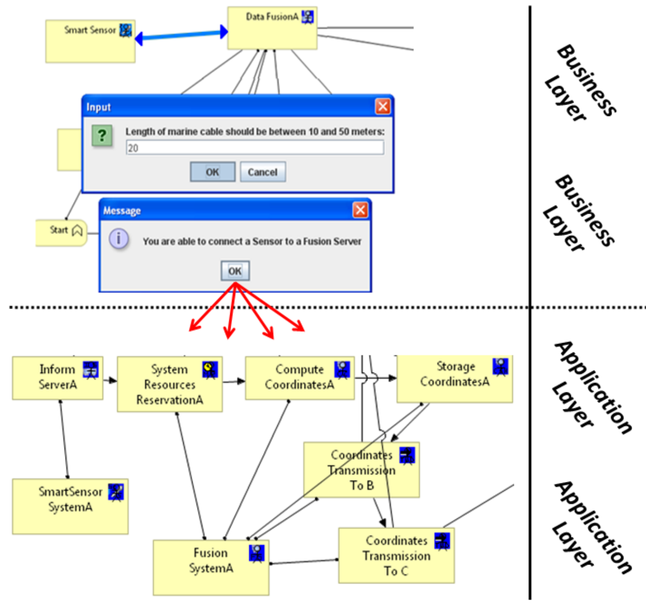

Fig. 10. Throwing MO concepts and relationships in the application layer after entering a proper value

standard, and classical design tool to have a specific one like ArchiMO; (C6) deploys different physical components (Sensors and Servers), and logical components such acquisition/localization algorithms.

VI. Object Localization Case Study

For assuring a continuous observation of the sea by MeDON, the infrastructure of these types of observations is based on localization systems. These systems mainly rely on approaches that use algorithms. Some localization algorithms for underwater environment are: Triangulation [37][38][39], Bounding-Box [37][38], set-membership [39], and Dive'N'Rise(DNR) [38]. All the mentioned underwater environment algorithms are compared to the triangulation and the result is less or more positioning accuracy which is the criteria of comparison. We are interested then for this criteria to select the algorithm that gives more position accuracy than the others which is the case of trilateration or multilateration. For simplifying the implementation of localization scenario, we select the simplest one which is the trilateration that is used by the most dominated positioning system in the world, the Global Positioning System (GPS) [40][5].

In order to validate our proposed tool, we used it to model the application of Object Localization using the different new elements, relations and constraints that are proposed in the Meta-Model (see Figure. 2). The design model was then applied to a model compiler (see Figure. 11) which we have developed in order 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.

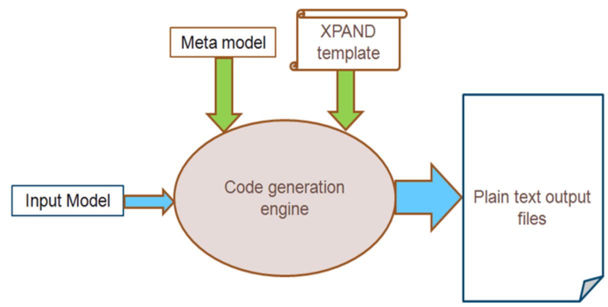

Fig. 11. The code generator workflow in XPAND language

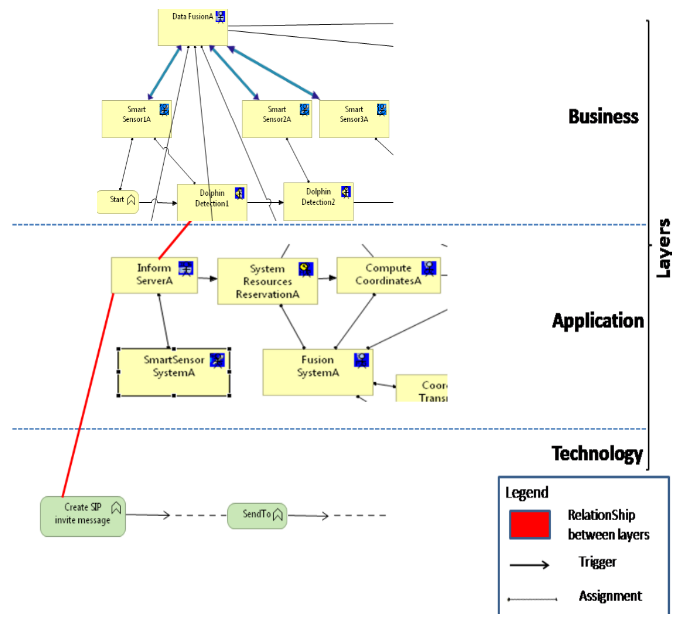

Fig. 12. Object Localization Underwater

A. Design Model

Using our generated design tool ArchiMO, we have modeled a system that localizes an underwater object. To localize this object, sensors should be connected to data fusion servers. We have applied the selected Data Fusion Architecture in section VI which is the Distributed for this design.

The design model is composed of three views regarding to the layers of ArchiMate (see Figure. 12): Business, Application, and Technology. In (Figure. 12), we present parts of the large model that is designed by ArchiMO. The model contains behavioral elements, in the business layer (see Figure. 12) shows the first activity of the smart sensor which is the dolphin detection1, etc. These activities are assigned to their proper smart sensors. Using the new extended SmartSensor and DataFusion Relationship these smart sensors are associated with the different data fusion servers and smart sensors that are required in the DFA

B. Compilation and Simulation

The design tool ArchiMO generates an XMI file to represent the graphical design, which helps conduct the design model

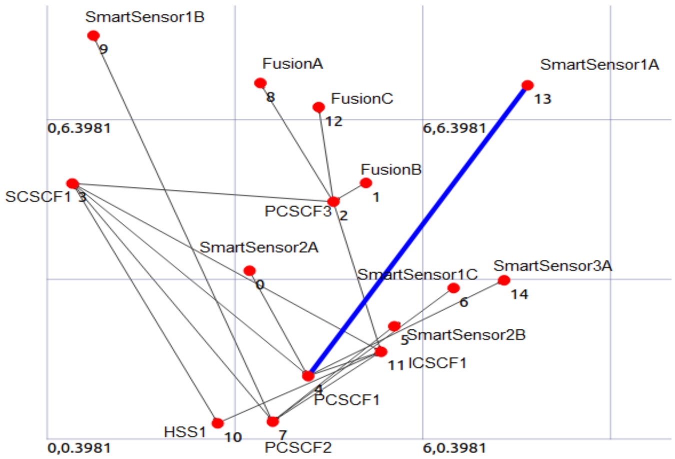

Fig. 13. Snapshot from the animation through NetAnim tool after running NS-3 simulation

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. 11). This hides complexity of constructing simulation programs from the designer and saves considerable time of the development process as it hides the specificities of the network simulator. The network simulator needs to be configured through a specific language and concepts using specific libraries that are not of the same abstraction level of the modeling tool and are not necessarily included in the design model. 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. 11) contains the mapping rules between the model elements and their representations in NS-3 [15].

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

(Figure. 13) shows the architecture of the system design that is generated by NS-3 for the mentioned design model. NS-3 generated 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 [14][15], and MO context [34][35]). The common design concept between all cases is the underlying platform (IMS) that represents the Platform Specific Model (PSM) [28].

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

We have presented in this paper a Mapping Approach between the different layers of EA for MO context. Our approach is based on extending the constraints of ArchiMate Meta-Model relying on MDE fundamentals. We have proposed a new design tool (ArchiMO) that is generated from the extended MO Meta-Model which considers the domain-specific concepts and constraints.

ArchiMO helps the designer in his concerned layer by getting concepts and relations automatically from the higher layer. ArchiMO helps the designer by avoiding making design errors earlier than verification and code generation activities. It assists also by the representing the domain-specific concepts of MO thanks to DSML fundamentals. We rely 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/instances.

By having mapping between layers, ArchiMO reduces the time of the design activity. And this is to throw quickly and instantly during the run-time of the design activity large number of specific MO concepts from a layer to another different layer instead of designing these MO concepts manually one by one by the other concerned designer in another viewpoint. Also, by having this mapping, the complex communication operation between the different designers according to their domain specificities is less complex. This is by reducing and minimizing the necessary exchanging knowledge between the designers team in order to design an overall integrated model. In other words, the designer of the lower layer just needs to provide once to the designer of the higher layer the corresponding related concepts or models that are specific for his layer according to each business activity in the higher layer. Once the designer of the higher level (Business Layer) finishes his business activities, the design of the application layer is mostly ready. It reflects less communication and negotiation than before between the designers.

Representing the connections logically between the different smart sensors and fusion servers helps to improve the understanding of the network constraints in a business level. Thus, it improves the communication of the design between the different stakeholders. Moreover, it provides a way to link between the design constraints and soft goals [41] as the design constraints are of a highly abstract level (business layer) and platform-independent.

On the other hand, 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. This extension could be the addition of specific constraints and features in the context of MO during the design activity.

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] N. Mitrou, *Networking 2004: Networking Technologies, Services, and Protocols; Performance of Computer and Communications Networks; Mobile and Wireless Communications; Third International IFIP-TC6 Networking Conference, Athens, Greece, May 9-14, 2004; Proceedings*. Springer Science & Business Media, 2004, vol. 3042.

[3] A. Cuzzocrea, *Intelligent techniques for warehousing and mining sensor network data*. IGI Global, 2009.

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

[5] C. Wang, *Localization and its applications in self-configurable wireless networks*. ProQuest, 2008.

[6] 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.

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

[14] 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. Vidács, Eds., vol. 7479. Springer Berlin Heidelberg, August 2012, pp. 100–111.

[15] 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.

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

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

[18] 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.

[19] 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.

[20] J.-L. Bakker and R. Jain, "Next generation service creation using xml scripting languages," *ICC 2002. IEEE International Conference on*, vol. 4, pp. 2001–2007, 2002.

[21] 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.

[22] W. M. Zimmer, *Passive acoustic monitoring of cetaceans*. Cambridge University Press, 2011.

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

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

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

[26] M. M. T. Zekai Demirezen, Barrett R. Bryant, "Domain specific modeling language design space analysis," in *UAB, Birmingham, AL 35294, USA*, June 2011.

[27] 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.

[28] 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.

[29] J. Loicq, L. Clermont, W. Dierckxb, T. Van Achteren, and Y. Stockmana, "A 100 m ground resolution global daily coverage earth observation mission," in *International Conference on Space Optics*, vol. 7, 2014, p. 10.

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

[31] E. Kaplan and C. Hegarty, *Understanding GPS: principles and applications*. Artech house, 2005.

[32] 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.

[33] I. Alloush, C. G. Aoun, Y. Kermarrec, and S. Rouvrais, "A domain-specific framework for creating early trusted underwater systems relying on enterprise architecture," in *Modelling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2014 IEEE 22nd International Symposium on*. IEEE, 2014, pp. 120–125.

[34] C. Aoun, I. Alloush, Y. Kermarrec, J. Champeau, and O. Zein, "Domain specific modeling language for object localization in marine observatories," in *SENSORCOMM 2014 : The Eighth International Conference on Sensor Technologies and Applications*, November 2014.

[35] J. Champeau and O. K. Zein, "A modeling approach for marine observatory," 2015.

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

[37] M. Erol, L. F. Vieira, and M. Gerla, "Auv-aided localization for underwater sensor networks," in *Wireless Algorithms, Systems and Applications, 2007. WASA 2007. International Conference on*. IEEE, 2007, pp. 44–54.

[38] M. Erol and Vieira, "Localization with dive'n'rise (dnr) beacons for underwater acoustic sensor networks," in *Proceedings of the second workshop on Underwater networks*. ACM, 2007, pp. 97–100.

[39] A. Caiti, A. Garulli, F. Livide, and D. Prattichizzo, "Localization of autonomous underwater vehicles by floating acoustic buoys: a set-membership approach," *Oceanic Engineering, IEEE Journal of*, vol. 30, no. 1, pp. 140–152, 2005.

[40] Z. Rothacher and B. Zebhauser, "Einführung in gps," *Tutorial zum*, vol. 3, 2000.

[41] S. C. R. H. Kenneth Boness, Marc Bartsch, "A practical approach to goal modelling for time-constrained projects," in *The University of Reading, Reading, Berkshire, RG6 6AY, UK*, 1999.