A Generalized Model Transformation approach to Link Design Models to Network Simulators
NS-3 Case Study

Iyas Alloush[1,2], Yvon Kermarrec[1,2] and Siegfried Rouvrais[1,3]
[1]*Telecom Bretagne, Institut Mines-Telecom, Université Européenne de Bretagne, Brest, France*
[2]*UMR CNRS 6285 Lab-STICC, Brest, France*
[3]*IRISA, Rennes, France*

Keywords: Model Driven Engineering, Simulators, NS-3, Telecom Service, Enterprise Architecture, Verification, Model Transformation, Code Generation, Tool Chains.

Abstract: Telecom service creation (TSC) activity is one of the most important phases of a TS life cycle. There are many efforts that were done to improve this activity recently. The early verification of the TS from its design models give an advantage to the service provider to improve the qualities and to detect design errors before the implementation phase. Simulation makes it possible to predict the system behavior avoiding the cost of real systems. Our objective in this paper is to present our methodology to link high abstract models of telecom services to network simulators. Relying on Model Driven Engineering, we propose a generalization of code generation methodology using an IMS meta-model and simulator-dedicated templates. In our approach, the network simulator specifications are related to the transformation template only, while the underlying platform specifications and standards are included in the meta-model. We illustrate our approach with a new transformation to generate configurations for NS-3. We apply an example of a video conference service to generate the simulation code.

1 INTRODUCTION

In the scope of the Telecom Service (TS) Verification (Combes and Renard, 1999), the service designer needs to verify the service design if it satisfies the predefined funtional and non-functional (Chung et al., 1999) requirements or not. This verification activity is better to be performed as early as possible, so to avoid the expensive consequences that may happen when correcting the design errors and flaws after the software and hardware installation.

We aim to assist the different stakeholders that are involved in the TS design (Combes and Renard, 1999), and more specifically, to help them detect the design errors and flaws, so to improve the qualities (Chiprianov, 2012) such as the performance, cost, etc.

Network simulators provide valuable feedback in the behavior of Telecom Services. Although, there are simulators which have friendly user interfaces and modeling environments, they still consume the designer time due to the complexity of the design architecture. The domain specificity of network simulators, the daunting coding, and long time configuration, makes it difficult to verify the TS through its design models manually.

The difference between the technologies that are used to design the telecom service architecture and the ones that are used to model it in the simulator environment raises a question: **How to link directly and automatically between these two technical spaces (design and simulation)?**

Observing the simulators' output makes it possible to check design errors and quality violations. Our objective in this paper is to provide the TS designer with a tool that links directly the design model to the simulator environment. Therefore, we can obtain specific measurements that will be needed in further verification activities according to the functional and non-functional requirements.

Our recent work (Alloush et al., 2012) illustrated our approach in using Model Driven Engineering (MDE) to link the high abstract models to simulators (e.g. OPNET). We rely on the core-network platform IP Multimedia Subsystem (IMS) to perform the video conference functions. IMS provides the ability of deploying large number of Telecom Services that

Alloush I., Kermarrec Y. and Rouvrais S..
A Generalized Model Transformation approach to Link Design Models to Network Simulators - NS-3 Case Study.
DOI: 10.5220/0004407503370344
In *Proceedings of the 3rd International Conference on Simulation and Modeling Methodologies, Technologies and Applications* (SIMULTECH-2013), pages 337-344
ISBN: 978-989-8565-69-3

are composed of different types of software application thanks to its advanced structure that mediates between different telecom evolutions. We have obtained manually the measurements that we relied on to check the behavioral view of the TS design. The design architecture of the TS was automatically generated by our tool relying on Model Driven Engineering (MDE) and Eclipse Modeling Framework (EMF) in Eclipse. The measurement settings were manually configured, and the measurement analysis were done manually to check the behavioral view of the TS design. Additionally, relaying on one simulator to test the design is not enough, as simulators differ from each other in their measurement capabilities and certificate levels.

In this paper, we generalize and develop our approach to be used with domain specific (network) simulators, and we include the automatic generation of the measurement configuration.

Our first contribution in this paper is in extending the Meta-Model (MM) that is presented in (Alloush et al., 2012; Chiprianov et al., 2011) to include new elements that improve the code generation abilities to adapt with different network simulators. We add new measurement and tool (simulator) entities to that MM to enable the automatic generation of the measurement configuration. We argue, as our second contribution, that by fixing the MM which represents the core-network platform (e.g. IMS), we can generate the code that is needed to configure different network simulators by changing the transformation template using Eclipse environment. We illustrate our approach with an example of configuration generation to the NS-3 network simulator, where as far as we know, there is no similar work done using NS-3 yet. This is added to the previously-argued case study in (Alloush et al., 2012) that is specific for another network simulator (OPNET) to illustrate the generality of our approach.

In section 2,we present briefly the related work, highlighting the points of interest to our objective. In section 3, we will provide an explanation about model transformations that we rely on to generate the code. In section 4, we highlight the Enterprise Architecture (EA) with its architecture, and show the benefits from applying its structure to our approach. In Section 5, we present a short explanation about network simulators, and show the features of the different simulators that we used in our work. Section 6 will be dedicated for our contributions, where we present our generalized approach to obtain wide range of measurements for the verification process. Then we present the new extensions of the Meta-Model, and we give a brief view to the NS-3 code generation method. In Section 7, we discuss an example of a session creation for a video conference TS, presenting the new model instances that are needed to generate the measurement configuration. Then we conclude in section 8 and discuss our future work.

2 RELATED WORK

This paper is in the scope of TS creation activity. As our work is a continuation to (Chiprianov, 2012) and according to the requirements of the early test activity, we will highlight some of the others work from this domain, taking into consideration the following criteria: (1) Involving the different actors during the service creation activity; (2) the Platform Specific Models (PSMs) level of details with structural elements and if it is enough to start simulation activity or not; (3) the ability to generate code directly from Platform Independent Models (PIMs); (4) The possible execution or simulation environments; (5) Usage of Domain-Specific tools; (6) The ability to collect measurements and to configure them automatically.

In the following, we present some of the work that is related to early verification activity before the implementation:

- In (Achilleos et al., 2008), the authors proposed an approach for service creation that introduces a service validation straight after the service design phase and before the implementation. Their methodology integrates the Model Driven Architecture (MDA) and Petri Nets to provide design, validation and code generation process.They use Petri Nets to validate the service behavior and ensure its correctness. Their approach involves the End-User in the final validation process, where the validation software is obtained by the same generative approach and MDA. Their methodology provides behavioral details only, and presents no relationships between different nodes of an execution network. This prevents the tester from being able to generate a network simulation scenario. They generate Java code for their example directly from the PIM interface model. The presented Meta-Model of Service Oriented Petri Net shows no possibility to analyze the behavior of the service using any other method else than Petri Nets;
- In (Hartman et al., 2007), the paper describes a model-based approach to create telecom services. They rely on the code generation to implement the service relying on IP Multimedia Subsystem (IMS) infrastructure. The stakeholder who is involved in their approach is the service provider.

They start from the high abstract models as PIM to define the service functions using state machine UML2 models. Using Eclipse software, they could close the abstraction gap between the design and the execution environment, by hiding the low level protocol and architecture details. The code generation process produces Java packages (e.g. SIP servlet) that can run on a SIP enabled Application Server (e.g. IBM WAS version 6.1). Simulation activity is not included as there is no hardware entities descriptions in the PSM;

- The work presented in (Adamopoulos et al., 2002) addresses the same scope of service creation activity. They propose a methodology that takes into consideration the customer and service provider requirements. The validation activity is performed after the implementation phase, and the service specification is represented by use-case models as PIMs. They apply code generation to produce Java and C++ codes directly from these PIMs. No simulators were used;
- The paper (Touraille et al., 2011) presents work that is very close to the formalism which we use. The authors apply MDE and rely on DEVS (Discrete EVent Systems Specification) formalism to smooth the modeling and simulation cycle and facilitate the coupling of models using heterogeneous formalisms. They provide the SimStudio software which aims at providing modeling and simulation tool chain. This assists the developer in all of the activities from the design to the result analysis. They produced an advanced tool chain that contains 4 axis: modeling, analysis, visualization, and Management. The SimStudio relies on DEVS that contains a simulation kernel which offers the possibility of high abstract level of modeling. Their approach and ours depend on meta-models that contains different views to the different activities of the tool chain (modeling, simulation). Although they have analysis tools that we didn't include in our tool chain, their tool chain doesn't have support for structural elements representation while limited to the behavioral ones. Their approach doesn't show domain specificity, where we rely on IP Multimedia Subsystem (IMS) platform. The different views of the actors of an enterprise are not considered as it shows in their approach.

In this paper, we will illustrate how our work satisfies all of the previously mentioned criteria.

3 MODEL TRANSFORMATIONS

Model transformation is one of the key concepts in MDE, and in our approach. There are different types of model transformations (van Amstel, 2010; Mens and Van Gorp, 2006) and they are categorized due to different views. There are different types of model transformations: model-to-model, text-to-model, model-to-text (which we use), and text-to-text. From the meta-model or grammar change view point, there are two types of model transformations: endogenous model transformation where the source and target models conform to the same meta-model, and the exogenous model transformation where the source and target models conform to different meta-models. Our model transformation is an exogenous model transformation, as the target is a configuration script for a simulator with different syntax rules and constraints. The difference in syntax and semantics between the two technical spaces (design and Simulation) creates a challenge to link between them. In our approach, the code generation rules are used to equilibrate this difference (Chiprianov et al., 2010).

XPAND is a model-to-text transformation language that makes it possible to iterate over input model instances to generate text files. It has the capability of collecting the data of the design model and insert it in the form of text embedded in the different statements of the code that is used to configure the target tool. This enables us to generate Java packages for high abstract models (PIMs) (Chiprianov, 2012), and configuration scripts for simulators for the platform specific models (PSM). We tend to make the XPAND templates as far as possible from the hard coding, this implies to define the needed entities in the Meta-Model and so in the input model.

4 ENTERPRISE ARCHITECTURE (EA) AND ArchiMate

An Enterprise Architecture (EA) "is an instrument to articulate an enterprise's future direction, while also serving as a coordination and steering mechanism toward the actual transformation of the enterprise" (Greefhorst and Proper, 2011). In our approach, we apply the EA standard into the TS architecture (Simonin et al., 2007). EA framework provides a telecom enterprise a way to decompose a complex architecture of a TS into aspects and layers. The aspects dimension provides an aspect conceptualization of an enterprise. While the layer dimension separates the TS architecture into 3 different layers according to the level of details.

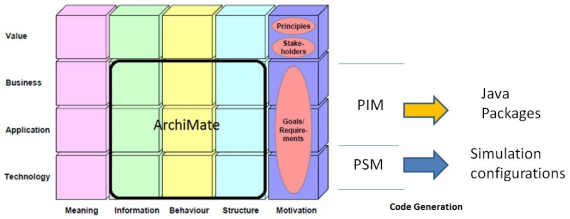

Figure 1: Enterprise Architecture and Code Generation in our approach.

In our approach, we apply ArchiMate which is an EA modeling language, as ArchiMate seems to be suitable for IT domain modeling (Chiprianov, 2012). Figure (Fig.1) presents the two dimensions of ArchiMate, and illustrates the separation in abstraction between the different layers from our view point. Regarding to the criteria in section 2, EA and ArchiMate provides a solution for the simulation needs, as it: (1) separates between the aspects of behavioral and structural entities; (2) provides designers with 3 layers that are different in the level of details, which makes it possible to specify the models of the technology layer as PSMs; (3) involves the different stakeholders in the design activity. This answers to the criteria 1, 2 and 3 that are mentioned in section 2. The dedication of layers due to the domains makes it possible to involve the different stakeholders of a service creation activity that is mentioned in (Combes and Renard, 1999). We consider the service provider, designer, and service developer stakeholders in our work, as an answer to the first criteria in section 2.

5 NETWORK SIMULATORS

System modeling makes it possible to modify the design parameters and to test the system and analyze it more than once, without adding experiment cost.

5.1 Network Modeling Approaches

To model a system for testing purpose, some simplifying assumptions are often required. Relying on too many assumptions for simplification purpose, may lead to an inaccurate representation of the system. In the scope of computer networks, there are two modeling approaches (Issariyakul and Hossain, 2009):

1. Analytical Approach. This approach relies on mathematical description for the system with the help of mathematical tools such as queuing and probability theories, then applying that description using the numerical methods. Such an approach may not give accurate representation of the real system if many simplifying assumptions were made;
2. Simulation Approach. This approach requires fewer simplifying assumptions (less abstract level in the model) than the analytical approach. One can set every needed specification in its place to describe the actual system in the best way.

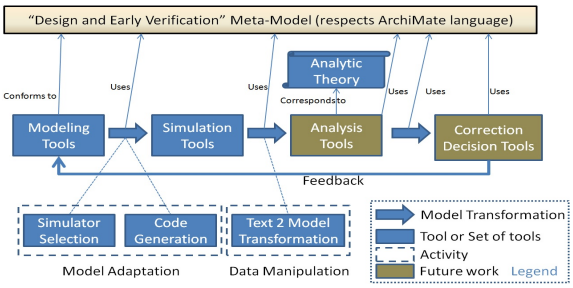

Figure 2: Our approach to verify Telecom Services after the design activity relying on tool chains.

Both approaches may leave out some details, since having a lot of details may result in large computational efforts and unmanageable execution. In our approach, at the design activity, we need to model architectures, with both structural and behavioral entities in high level of abstraction. Thus, we have chosen the simulation approach for modeling the TS design, as we prefer to have a better description of the TS design and to execute its behaviors in an accurate way.

5.2 NS-3 Simulator

We select NS-3 as it: (1) supports tracing functions, to collect the measurements and export them into files; (2) accepts configuration using Python and C++ for front-end (e.g. scripting, visualization); (3) supports different types of easy-configured applications (e.g. Socket using TCP or UDP). NS-3 is a proper simulator that fits the nature of TSs, as it accepts (in addition to the previous features) plain text input files, and generates measurements that can be analyzed to improve the qualities of the TS.

6 CONTRIBUTIONS

6.1 Methodology

Our general approach relies on tool chains to achieve a complete loop 2 between the design and early verification activities in Telecom Service Creation (TSC) activity (Combes and Renard, 1999).

The verification activity may require different simulators according to the requirements nature and

variety, where the TS is going to be verified with respect to these requirements. Therefore, we aim to provide the designer with a methodology that is able to use wide range of simulators. In this paper, our contributions have one broad aim (Fig.3) : to link the high abstract design models from the TS domain to the network simulators so to obtain measurements from the modeled system and analyze in the next activities. Therefore, we highlight two major contributions that helps achieving this target:

- Extending the technology layer meta-model that is mentioned in (Alloush et al., 2012) to include new adequate entities that are relevant to the simulation activity and of general usage for different types of network simulators;
- Proposing an approach based on MDE to link between design models and simulators using a generalized method, we fix the Meta-Model (syntax) and change the model transformation template only to generate the code for the different simulators. This is coherent with the advanced core-network platforms (e.g. IP Multimedia Subsystem IMS (Camarillo and García-Martín, 2008)), as changing the infrastructure for every new IT service is not a practical or economic solution. Normally, once the infrastructure is installed and implemented, it is better for the network provider to invest in it for long period.

6.2 Meta-model Extension

We present the technology layer Meta-Model extension that is needed to generalize our approach to link between the high abstract models and the simulators. The extension is tool (simulator) oriented, and includes the following actions:

1. Extending the NodeInterface entity (Fig.4) which is defined using ArchiMate language (The Open Group, 2009) and of the type InfrastructureInterface. We have defined more protocols that correspond to the OSI (Open Systems Interconnection) system. This makes it possible to map the design with different types of network simulators of different granularity levels;

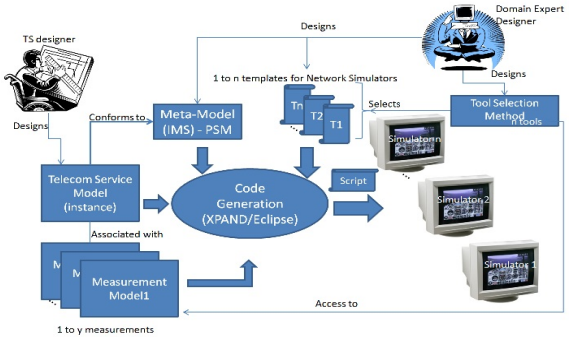

Figure 3: An MDE approach to link high abstract models to simulators including measurement configuration.

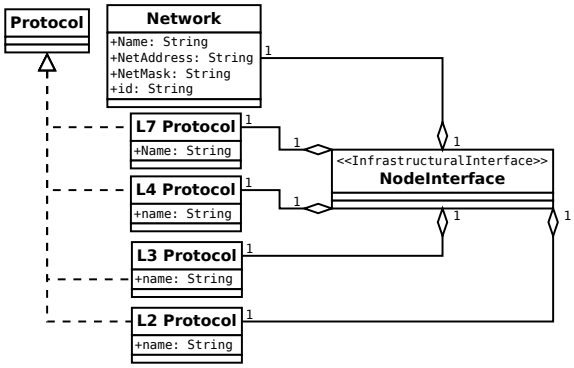

Figure 4: NodeInterface extension to the IMS meta-model presented in (Alloush et al., 2012).

2. We define the Measurement model (Fig.5) of type TechnologyLayerElement from ArchiMate language (The Open Group, 2009) that contains: measurement function, measure, and the probe. The probe (Chang, 1999) is the entity that is related to the simulator part, while the measure is the register that saves the final result of the data manipulation after the call of the measurement function. To enable the manipulation of the collected data by different probes, we define mathematical operators such as (Sum, Subtract, Multiply, and Divide, etc). These operators are used in the measurement function to enable the customization of measurements by the domain experts (Fig.3). These measurements are linked to the design entities (behavioral and/or structural) through association relations (The Open Group, 2009). The reason behind this link is that there are some measurements related to hardware elements (e.g. CPU utilization is related to nodes) while (Session Start/End Probe is related to a behavior of exchanging control messages between nodes). This contribution answers to the criteria 6 in section 2;
3. We define the "associated with" relationship from ArchiMate language (The Open Group, 2009) to associate structural and behavioral entities of an IMS platform with the corresponding measure as a result of the measurement function that is processed in the simulator.

The mentioned IMS Meta-Model and its extensions are prototyped and implemented in an ecore file (Meta-Model) using Eclipse EMF tool.

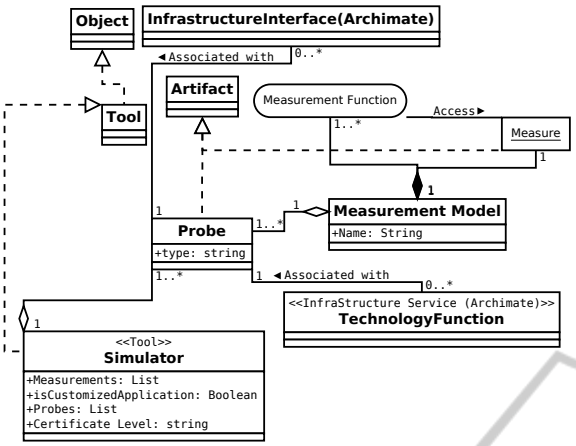

Figure 5: Measurement view from the tool specification Meta-Model linked with IMS MM presented in (Alloush et al., 2012).

6.3 Using Multiple Network Simulators

The extended meta-model presented in subsection 6.2 have the same specifications that were used in the previous XPAND template that is used with OPNET tool in (Alloush et al., 2012), and it works now also with the NS-3 simulator template, where we use the same model transformation language (XPAND) in section 3. OPNET and NS-3 are widely used in the domain of network simulation, and the extended MM provides enough abstract syntax to implement and run the model transformation. Additionally, the extension depends on telecom standards and basics that every network simulator should respect. This enables us to answer the question of reusing our approach with other network simulator.

```
PointToPointHelper p2p;

«FOREACH this.eRootContainer.eAllContents.typeSelect(CommunicationPath) AS CP-»
«REM» The Communication path starts the iterations to define the interfaces types and their atttributes
    NodeContainer N_«CP.name.replaceAll("[ ]+","").replaceAll("[-][/]","_")»;
    //******Here, we can define the predefined QoS attributes of the P2P link between the two nodes****
    NetDeviceContainer dev_«CP.name.replaceAll("[ ]+","").replaceAll("[-][/]","_")»;
    «LET {} AS interfacelist-»
    «LET {} AS InterfaceSourceNodeList-»
    «LET {} AS nodepair-»
    «LET {} AS nodepairid-»
    «FOREACH this.eRootContainer.eAllContents.typeSelect(AssociationRelationship) AS e-»
        «IF e.target.id.matches(CP.id)-»
            «interfacelist.add(e.source.name).toString().replaceAll(".*","")-»
        «ENDIF-»
    «ENDFOREACH-»
    «FOREACH interfacelist AS e-»
        «FOREACH this.eRootContainer.eAllContents.typeSelect(AggregationRelationship) AS ee-»
            «IF ee.target.name.matches(e.toString())-»
            «InterfaceSourceNodeList.add(ee.source.name).toString().replaceAll(".*","")-»
            «ENDIF-»
        «ENDFOREACH-»
    «ENDFOREACH-»
```

Figure 6: Part of the XPAND template presents the iterations to decompose and map the Service Design to NS-3 model.

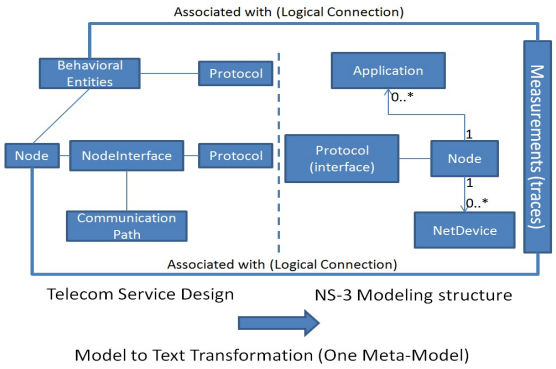

Figure 7: Mapping elements between TS design and NS-3 technical spaces (implemented in XPAND).

6.3.1 Mapping TS Design Model to NS-3 Model

We will present the main conceptual modeling concepts in NS-3 and its hierarchy, and how we mapped it (Fig.7) to the IMS models that we have designed. To build up the topology of the network and the service structural design, we use the structural view from the model of the TS instance. This model contains the structural entities of the TS service, one of these entities is the CommunicationPath that represent the point-to-point connection between to NodeInterfaces. Thus, iterating for every CommunicationPath (Fig.6) makes it possible to identify the parent NodeInterfaces, and then identifying the nodes that are joined together using this CommunicationPath (Fig.10).

7 VIDEO CONFERENCE CREATION EXAMPLE USING IMS CASE STUDY

In this section, we illustrate our approach with an example of a TS (Video Conference) relying on IMS network platform (Camarillo and García-Martín, 2008). We are interested in the control plane (signaling) messages that are exchanged between the different nodes to perform the creation of the conference, and we focus on the application layer messages. This is an important point for the designer. The Session Initiation Protocol (SIP) (Camarillo and García-Martín, 2008) is the main application protocol in IMS. SIP messages are text plain messages, the fact that makes it easier to read and analyze them.

7.1 TS Design Model Instance (Static View)

The design model of the TS (right part of Fig.10) contains high abstract level of details such as Nodes, application protocols, messages, sequence of functions that call each other, etc. We present the structural view of the design and from the technology layer specifically. The CommunicationPath entities are clear in this example, we use them as a starting point to identify the structure of the simulation scenario as mentioned in 6.3.1. Relying on the types defined in the MM and the relations from ArchiMate, we can investigate the topology of the scenario using the iterative functions in XPAND (Fig. 6).

7.2 Examples of Measurements and NS-3 Code Sample

Measurement instances should conform to the meta-model (Fig. 5). For an IP network, we present 2 examples of measurement models that will be implemented in NS-3 code automatically.

- DevQueueDrop: This is an example for a customized measurement. This event is triggered when a packet is dropped on the device level in NS-3. We want to implement the trace of Queue Packet Drop by defining a counter that increments by 1 every time the probe is triggered (Fig.8). This mathematical operation is implemented in the measurement function, where the inputs are: the probe value itself and the constant 1;
- CongestionWindow: This measurement exists in the NS-3 simulator, and their is no need to implement a measurement function for such a built-in measurement (Fig.9).

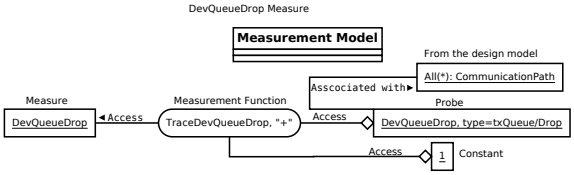

Figure 8: A model instance for DevQueueDrop Measurement.

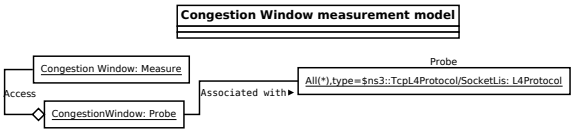

Figure 9: A model instance for CongestionWindow Measurement.

Unlike OPNET, the configuration method in NS-3 (NS3, 2012) makes it possible to use different types of variables to be used inside the code for many purposes such as calculations.

(Fig.10) presents the correspondence between the generated code and the structural entities. Using (waf) command we verified the result code with no errors in the C++ rules neither in the NS-3 ones.

8 CONCLUSIONS AND FUTURE WORK

In this paper, we have presented our approach to connect the design models from the specification activity to the network simulators directly. A Meta-Model extension was proposed to generalize the linking between the modeling and network simulation technical spaces relying on the reusability of the extended MM. We rely on the MDE approach to keep the abstraction level high at the design phase and to hide the complexity from the designer. To verify the telecom service design early at the specification phase, our approach joins the two different technical spaces (network simulation and TS design) together, attaching the measurement entities that exists in the network domain directly to the high abstract modeling tools. Thus, we put the verification utilities between the TS designer hands, taking the tool specificity and the daunting coding away from his responsibilities. We have illustrated our approach with some examples from the generated code and the corresponding input models.

Our approach separates between the roles of the technical spaces mapping, where the meta-model is reusable with different types of network simulators and the xpand template is related to the simulator specifically. The usage of multiple simulators enable the verifier from testing the TS using wide range of measurements. On the other hand, our method in generating the customized measurements takes into consideration only two inputs for the measurement function element. Additionally, the implementation of the XPAND template needs experience in the target simulator and modeling technical spaces besides to the IMS platform ”execution platform”. These limitations should be taken into consideration by the tool vendor when adding new tools.

In the future, on the one hand, we intend to define the method of tool selection. On the other hand we will integrate between these tools so the data flows from one tool to another, this makes the tools complement each other in our tool chain.

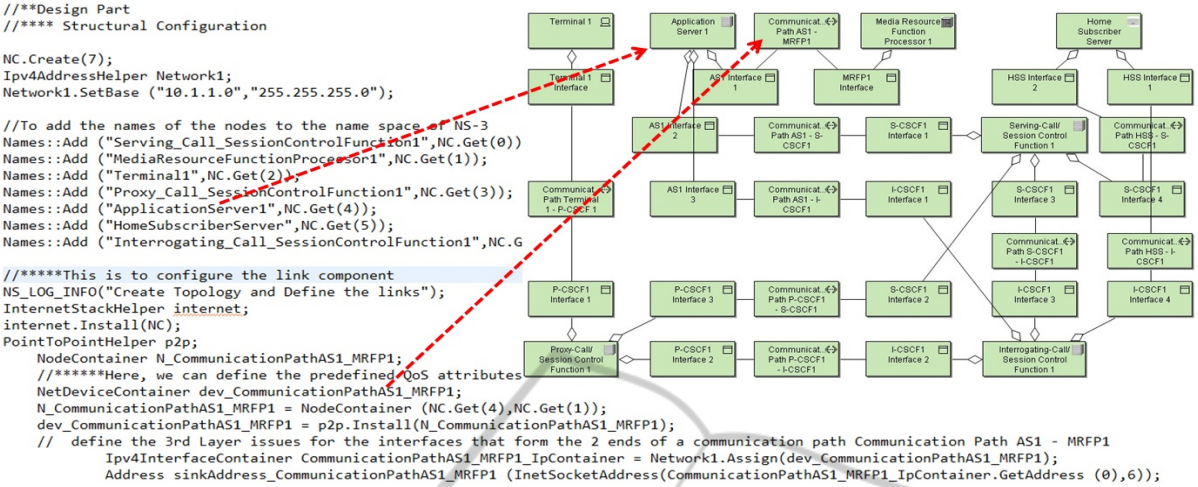

Figure 10: C++ code sample as input of NS-3 mapped to the design model.

REFERENCES

(2012). *NS-3 Manual-Release ns-3-dev, www.nsnam.org/docs/manual/ns-3-manual.pdf*, dev edition.

Achilleos, A., Yang, K., Georgalas, N., and Azmoodech, M. (2008). Pervasive service creation using a model driven petri net based approach. In *Wireless Communications and Mobile Computing Conference, 2008. IWCMC '08. International*, pages 309 –314.

Adamopoulos, D., Pavlou, G., and Papandreou, C. (2002). Advanced service creation using distributed object technology. *Communications Magazine, IEEE*, 40(3):146 –154.

Alloush, I., Chiprianov, V., Kermarrec, Y., and Rouvrais, S. (2012). Linking telecom service high-level abstract models to simulators based on model transformations: The ims case study. In *Information and Communication Technologies*, volume 7479, pages 100–111. Springer Verlag.

Camarillo, G. and García-Martín, M. A. (2008). *The 3G IP Multimedia Subsystem (IMS) Merging the Internet and the Cellular Worlds*. Third edition.

Chang, X. (1999). Network simulations with opnet. In *Simulation Conference Proceedings, 1999 Winter*, volume 1, pages 307 –314 vol.1.

Chiprianov, V. (2012). *Collaborative Construction of Telecommunications Services. An Enterprise Architecture and Model Driven Engineering Method*. PhD thesis, Telecom Bretagne, France.

Chiprianov, V., Alloush, I., Kermarrec, Y., and Rouvrais, S. (2011). Telecommunications service creation: Towards extensions for enterprise architecture modeling languages. In *6th Intl. Conf. on Software and Data Technologies (ICSOFT)*, volume 1, pages 23–29, Seville, Spain.

Chiprianov, V., Kermarrec, Y., and Rouvrais, S. (2010). Meta-tools for Software Language Engineering: A Flexible Collaborative Modeling Language for Efficient Telecommunications Service Design. In *Flex-iTools2010 ICSE Workshop on Flexible Modeling Tools*.

Chung, L., Nixon, B. A., Yu, E., and Mylopoulos, J. (1999). *Non-Functional Requirements in Software Engineering*, volume I. KLUWER ACADEMIC Publishers.

Combes, P. and Renard, B. (1999). Service validation. *Computer Networks*, 31(17):1817 – 1834.

Greefhorst, D. and Proper, E. (2011). *Architecture Principles*, volume 4 of *The Enterprise Engineering series*. Springer.

Hartman, A., Keren, M., Kremer-Davidson, S., and Pikus, D. (2007). Model-based design and generation of telecom services.

Issariyakul, T. and Hossain, E. (2009). *Introduction to Network Simulator NS2*. Springer.

Mens, T. and Van Gorp, P. (2006). A taxonomy of model transformation. *Electron. Notes Theor. Comput. Sci.*, 152:125–142.

Simonin, J., Le Traon, Y., and Jezequel, J. M. (2007). An enterprise architecture alignment measure for telecom service development. *11th Ieee International Enterprise Distributed Object Computing Conference, Proceedings*, pages 476–483.

The Open Group (2009). ArchiMate 1.0 Specification.

Touraille, L., Traoré, M. K., and Hill, D. R. C. (2011). 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*, pages 229–237, San Diego, CA, USA.

van Amstel, M. (2010). The right tool for the right job: Assessing model transformation quality. In *Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual*, pages 69 –74.