An Automated Tool Selection Method based on Model Transformation: OPNET and NS-3 Case Study

Iyas Alloush, Yvon Kermarrec, Siegfried Rouvrais

An Automated Tool Selection Method based on Model Transformation: OPNET and NS-3 Case Study

Iyas Alloush
Telecom Bretagne
Institut Mines-Telecom
Université européenne de Bretagne
UMR CNRS 6285 Lab-STICC
iyas.alloush@telecom-bretagne.eu

Yvon Kermarrec
Telecom Bretagne
Institut Mines-Telecom
Université européenne de Bretagne
UMR CNRS 6285 Lab-STICC
yvon.kermarrec@telecom-bretagne.eu

Siegfried Rouvrais
Telecom Bretagne
Institut Mines-Telecom
Université européenne de Bretagne
IRISA
siegfried.rouvrais@telecom-bretagne.eu

***Abstract*—Errors in telecom service (TS) design may be expensive to correct by telecommunication enterprises, especially if they are discovered late and after the equipments and software are deployed. Verifying complex architectures of TS designs is a daunting task and subject to human errors. Thus, we aim to provide supportive tools that helps during the TS creation activity. Network simulators play an important role in detecting design errors and predicting performance quality violations in the TS domain due to the measurements that they can produce. The metrics associated with performance requirements are numerous, and it is difficult to find a unique tool that can handle the prediction of their values. In this paper, we tackle the tool selection challenge for non domain-expert designers taking into consideration the differences between tools and the large number of metrics that they can measure. Thus, by applying model transformation techniques, we propose a method to select the proper tool(s) to obtain the measurements needed during verification activity. Therefore, we present our contributions on the modeling language level, and the tool selection algorithm with its implementation. Reusability, complexity, and customized measurements are taken into account. We illustrate our approach with a video conference and customized measurement example using OPNET and NS-3 simulators.**

I. Introduction

In the large telecommunication markets, with the far distances and increasing numbers of users, the telecommunications economy is growing rapidly. This market is extending, due to the large number of customers and to the increasing need to customized services.

These telecom services (TS) should serve the customers who acquire services with high performance requirements. An enterprise need to launch TSs with good quality rapidly before the other competitors. The TSs have specificities that make them different from other services or software. Thus, a specific life-cycle was introduced in the 90's [1], [2]. A TS is characterized by its functional (FR) [3] and non-functional requirements (NFR) [3]. The design activity plays a fundamental role in the TS life-cycle, as it is the activity where the service requirements are translated into specifications. Errors in TS design may be expensive to correct, especially if they are discovered late and after the equipments and software are installed. Therefore, the TS should be verified earlier at the design phase to check if these requirements are satisfied or not, so to improve the qualities and to correct the detected errors as early as possible. **Verification [4]:** "In design and development, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity".

A recent PhD [5] proposed a process of TS construction, while giving the opportunity to improve the QoS and to involve the customer in that process. In this dissertation, the author has shown that a design complexity can be reduced by the distribution of roles between the different stakeholders, so they share the design responsibilities each due to his domain experience thanks to the Enterprise Architecture (EA) and model driven engineering (MDE) principles [6]. The approach in [5] defines the requirements that the TS verification activity after the design phase should respect:

- Domain Specificity through the re-use of existing telecom-dedicated network simulators;
- Rapid prototyping as the verification activity is right after modeling activity and before implementation [1];
- Early verification [7], [8];
- More easy evolution of services as the transformation from Domain Specific Modeling Languages and the network simulator is automatic.

The proposal of this paper is a continuation of that dissertation in the design phase [5], so we propose our new activities in the same scope but as another activity in the service life-cycle: the verification one. In our research, we aim to provide the different stakeholders with supportive tools that helps them during the design activity of a TS.

A TS may use several applications that interact with each other to perform the service functions. These applications are different themselves in the resources they reserve. Some applications are sensitive to a specific group of metrics that differs from other applications. For instance, jitter measurement is needed when the service contains a video application, while it

is not important for an FTP application. Besides, the types of measurements are different in their natures and domains (e.g. performance, security, etc). To detect errors and performance quality violations in the telecommunication domain, network simulators plays an important role in the telecommunications domain. They can provide the verifier with the different measurements that makes it possible to estimate the TS performance. The metrics associated with performance non-functional requirements are numerous. Therefore, one tool (e.g. OPNET or NS-3) may not be able to cover all of the measurements needed in the verification activities for all kinds of services and requirements. The challenge that we face in this paper is: ***How to tackle the tool selection challenge for non domain expert designers taking into consideration the differences between tools and the large number of metrics that they can measure?***

Our objective in this paper is to propose a method for tool selection, that is based on the MDE principles. The selected tool(s) will form a tool-chain that integrates results together so to have a wide range of measurements in a specific domain (e.g. Network Domain). Tools differ from each other due to different properties such as their domains, capabilities such as measurements, and certification level. In this paper, we consider the measurement capability only, while one may model other specifications such as certification level of the tool.

Our approach is based on MDE principles that offers automation using model transformation techniques and deals with models to exchange data between the different activities. Model transformations help on the first hand in handling the large number of measurements that are required, and in helping the designer to find the proper tool(s) without the need to domain experience on the other hand.

In this paper, we present two contributions that are done to answer our research challenge. The first contribution is to define the modeling language entities, rules, and constraints using Eclipse Modeling Framework (EMF) tools that are available in the Eclipse IDE, this contribution serves the purpose of this paper and other activities in our general approach. Our second contribution is in the algorithm of the tool selection method where we implement it using a model transformation language XPAND [9]. The key reason behind using model transformation is to mix the modeling and automation concepts together.

Earlier, we used only one tool (simulator) to validate the TS design described in [5], where we depended on measurements obtained from OPNET simulator [10], [11]. Recently, we have implemented another model transformation to link the high abstract models to simulation level (NS-3). Thus, we have a tool-chain that contains different simulators (OPNET [10], and NS-3). This forms a case study for our proposal in this paper.

Our method is applied using Eclipse Modeling Framework (EMF), where it depends on models and transformation. The meta-model provides the types and constraints that are needed in the transformation and the measurement modeling. The reusability of this method comes from the possibility of changing the meta-model due to the domain specifications, while the algorithm stays unchanged. Implementing the algorithm using XPAND language makes the tool selection activity homogeneous with the other model adaptation activities (Fig.3).

On the other side, modeling the measurements and tool specifications is an action that needs domain, modeling, and tool specification experience.

In section 2, we present related work and highlight the criteria that distinguish our work. Section 3 presents notions of FRs and NFRs with a simplified example from the telecommunications domain. In section 4, we present the EA standard and how it is applied in our approach. Section 5 presents our approach activities and highlights the model adaptation one that includes tool selection method. We explain in section 6 our proposed method for the tool selection. In section 7 we present our contributions to achieve the tool selection method, describing its mechanism. Section 8 illustrates our proposal by an example from the video conference instance that we have implemented in a previous work. Finally, we conclude and present our future work.

II. Related work

In the scope of verification of system architectures from the design models, we present related work taking into account the following aspects:

1) Reusability of method in the same domain;
2) Are there Intermediate Models between design and analysis activities?;
3) Selecting automatically between tools due to measurement capability;
4) High level of abstraction for modeling;
5) Linking Measurements with Design Level for verification purposes.

- In [12], the authors present a way to verify the performance properties of coordination models, by extending the Reo coordination model with stochastic properties "Stochastic Reo". They use an intermediate model transformation to get the QIA models which are extensions of Constraint Automata. With the Stochastic Reo they can specify the behaviors of connectors, and the behavior of the environment. Their method shows reusability in the domain of Embedded Systems. Many models are needed to generate the configuration of the verification tool (PRISM). Their tool-chain enables the designer to stay on a high level of abstraction during the design activity, but it can't generate measurements in relation to the non-functional requirements. The measurements that it produce are built in the verification tool and there is no ability to define a customized measurements;
- The authors in [13] worked on analyzing the behavioral models of Embedded Systems using tool-chain for the REMES modeling language. The REMES editor gives a high level of abstraction for modeling behaviors. A direct automated transformation from REMES to Priced (Timed Automata) is done to perform the formal analysis. The analysis were done in relation to the functional

and non-functional requirements. Their approach is able to make simulations and obtain statistics to perform formal analysis on the system's model against the various requirements. Their method shows no automated tool selection techniques. Intermediate model transformations were needed to bridge the system design to the tool-chain. This method does not show a feature of customizing measurements;

- In [7], the authors proposed the Model Driven Analysis approach, where an Intermediate Constructive Model (ICM) is generated from the design model to link between the design and analysis activities. Their approach provides the capability of high abstract modeling, and gives the ability to analyze the design due to the functional and non-functional requirements through different tools and using intermediate model transformations. In this approach there is no mention about how to select the proper tool(s) for verification purpose;
- The authors in [14] use the FIACRE language as a pivot language in their model checking approach in the domain of Embedded Systems. Their objective is to perform behavioral verification of the Embedded System models designed using AADL. This method depends on intermediate model transformations to move from tool to another, so many languages and interpretations are used for this purpose. They can make behavioral verification using the AADL, FIACRE, and Tina tool-chain, depending on Timed Transition Systems (TTS) which is an extension of Time Petri-nets that handles the data and priorities. The model transformations that are used in their approach enable them to use different tools, but no method to choose between the tools directly from the statistics that they can provide;
- In [15], the authors present an approach to link between the design and analysis technical spaces. They propose using AADL as a pivot language, so to provide a link between Stood design tool and Cheddar for the real-time analysis. Their method offers the capability of making feasibility tests on the system design represented using hardware and software entities. Cheddar can provide the evaluator with the capability to make performance analysis. This approach provides the designer with a high level of abstraction to design the system, while it does not select the tool(s) directly from the measurement needs or other tool specifications.

Not far from this related work and in the scope of early verification of TSs through their design, we propose a method that selects the proper set of tools, thus to select the right template of the code generation that is needed to generate automatically the simulation configuration script. This leads to link between the design and verification activities. In this paper, we will present our proposal to select the proper tool(s) according to the measurement capability.

III. Functional and Non-functional Requirements

Functional Requirements (FR) [3]: They are the software requirements that defines the behavior of the system. For instance, the behaviors that were defined in the different layers of the TS architecture in [16] present the functional requirements of the Video Conference service.

Non-Functional Requirement (NFR) [17]: is an attribute of or a constraint on a system. Non-functional requirements can be described using Softgoal Independent Graphs (SIG). The term (soft) comes from the nature of these goals. Softgoals are usually associated with labels that identify the degree of their achievement.

An evaluation process is needed to determine whether the softgoals are achieved or not, by setting or modifying their labels through labeling algorithm. The relationship which links between the softgoals and the functions is called the contribution. We give an example of a performance NFR for a TS: "The service of Video Conference should be delivered to 1000 users maximum simultaneously at the quality level A". In this example, one can notice that there is a clear relationship between the TS and the quality of service. In our approach, we take quality of service (QoS) into consideration during the verification process.

IV. Enterprise Architecture and Modeling of TSs

A. Enterprise Architecture

In our approach, we apply the EA standard into the TS architecture [6]. EA framework provides a telecom enterprise with 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 due to the abstraction level.

We apply Archimate [18] which is an EA modeling language. This language takes three design aspects into consideration: Information, Behavior, Structure. Thus, the entities in our models are classified according to these aspects. It decomposes the design abstraction into 3 layers: Business, Application, Technology (Fig.1). This answers to the criteria (4) that is

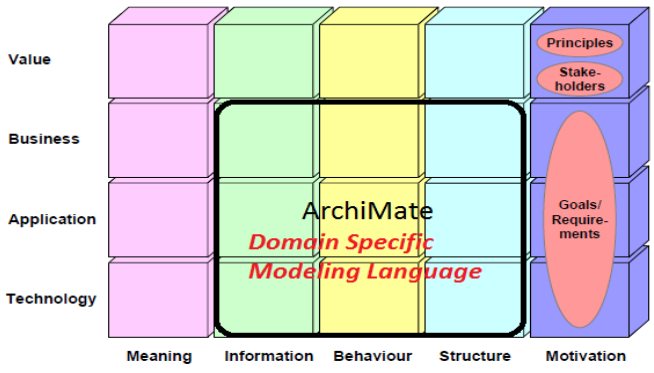

Fig. 1. Enterprise Architecture and Archimate [19]

mentioned in section II. The architecture is divided between

different levels of abstraction in a way that makes it possible to share the views of different stakeholders of an Enterprise. The proposed Domain Specific Modeling Language (DSML) in [5], [20] applies the mentioned EA standard and MDE approach, this makes it possible to hide complexity from the TS designer. We use the same language by extending the Meta-Model (MM) that was defined, and adding new verification entities and relationships. The new extended MM defines an extended language that has the same characteristics in relation with criteria (4) in section II.

V. Model Adaptation Activity

In our approach, we divide our verification activity into 4 main sub-activities (Fig.2):

- Linking activity: to link the softgoals that are defined in the requirements phase to the design models that are produced in the design phase.
- Model Adaptation activity: to select the right tool and then to transform accordingly the design models into a configuration script(s) that will configure the tool(s).
- Measurement Analysis activity: to analyze the measurements that are obtained from the tool-chain relying on an analytical theory (e.g. Queuing Theory), so to verify whether the TS design satisfies the functional and non-functional requirements (e.g. Performance) or not.
- Feedback activity: to benefit from the verification results and correct the design flaws, errors, and quality violations.

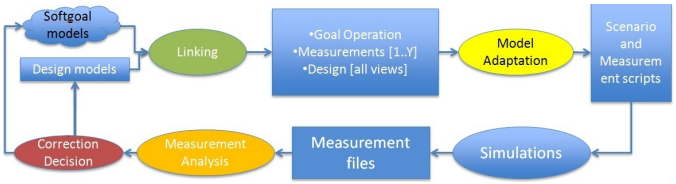

Fig. 2. Verification Activities in our approach

The model adaptation activities (Fig.3) are related to the linking activity in our approach, as there are different types of models and meta-model views that are designed in the linking activity. The tool selection has to decide which tool or set of tools are needed to obtain the measurements that correspond to specific requirements.

VI. Tool Selection Method

A. Model Transformations

Model transformation is one of the key concepts in MDE, and in our approach. There are different types of model transformations and they are categorized due to different views [21] such as: model-to-model, text-to-model, model-to-text, and text-to-text. We use a model-to-text transformation in our work to generate codes for simulators. We choose the same transformation language that is used to generate the simulation code to implement our method. This keeps the homogeneity between the different activities in our approach.

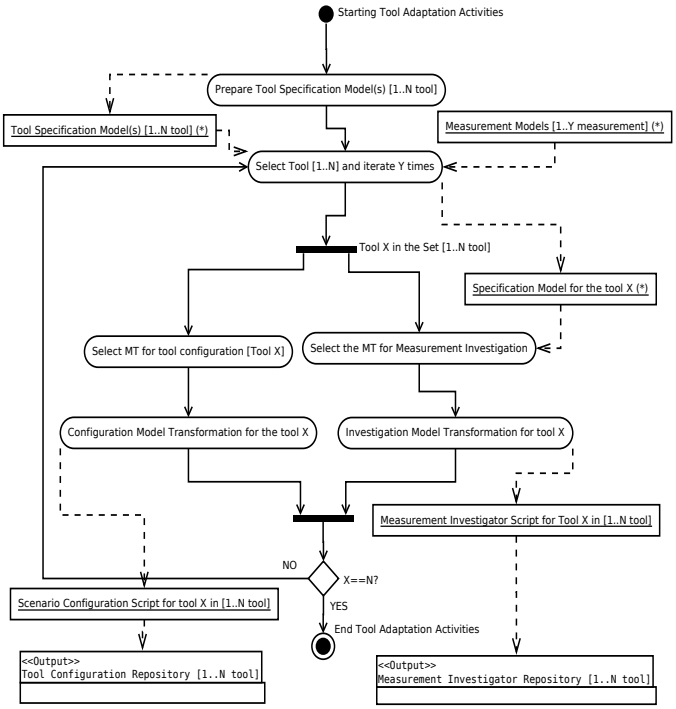

Fig. 3. Adaptation activities (UML activity diagram)

B. XPAND Model Transformation language

XPAND [9] is a model to text transformation language. It was originally developed as part of openArchitectureWare (oAW) project before it became a component under eclipse. XPAND includes an editor which provides features like syntax coloring, error highlighting, navigation, refactoring and code completion. XPAND functions has the capability of querying the information from the input model and generating text files that include the processed data (Fig.4).

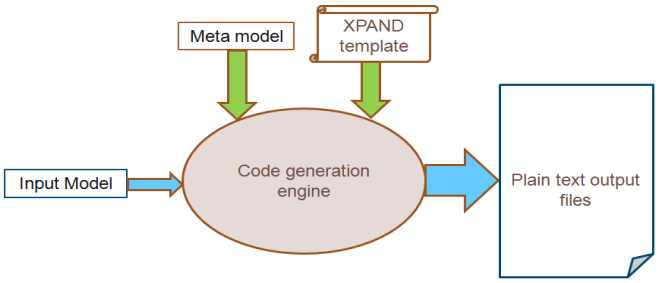

Fig. 4. XPAND Work-flow

C. General Method

The tool selection activity is one of the Model Adaptation activities, where the decision of which tool to generate the simulation code for, is needed. The selection of the proper tool implies to the selection of the code-generation template accordingly (Fig.4).

We apply MDE approach, so every data representation is model-based. Thus, the inputs of our proposed method are all models. These models are: Tool Specification models, Measurement models, and the Design-Verification Meta-Model. The designer can add another Tool Specification such as Certification Level, and so he needs to represent the Certification Level as a model.

This means that a **specification in the Tool model at the tool modeling dimension turns to a model or class at the dimension of the specification modeling, and the selection algorithm is going to check the tools according to the model of the required specification** (Fig.5) [1]. The filtration (tool selection) method is implemented using iterations of a model transformation language (e.g. XPAND). This makes this activity homogeneous with the other activities of code generation to configure the simulators directly and using one model transformation, as shown in [10]. The mentioned method answers the criteria (1, 2, 4, 5) in section II. Additionally, our proposed method can be applied to other domains, by using other meta-models and models of different entities' nature.

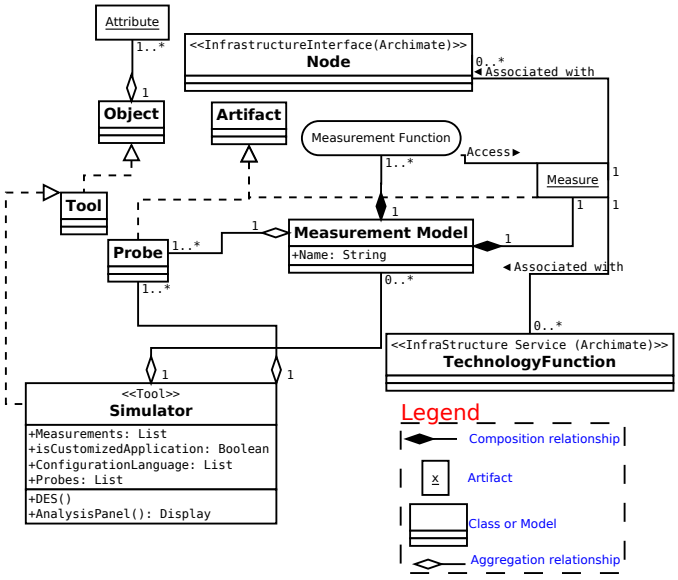

Fig. 6. Measurement & Tool View from the Extended Meta-Model

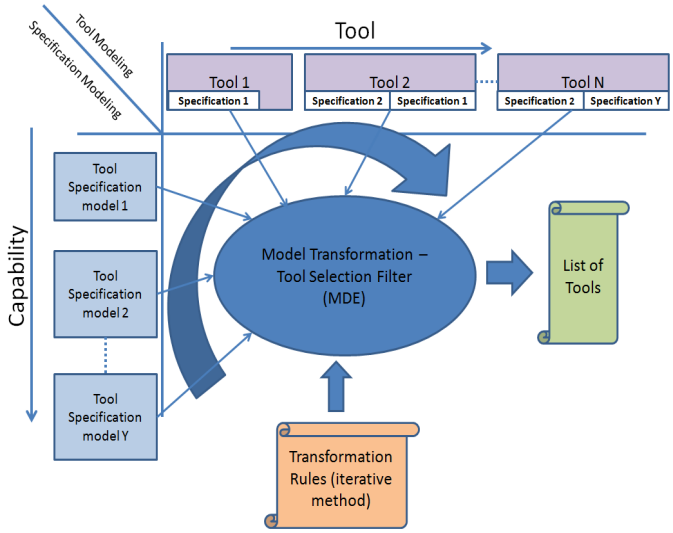

Fig. 5. Tool-Selection method based on MDE

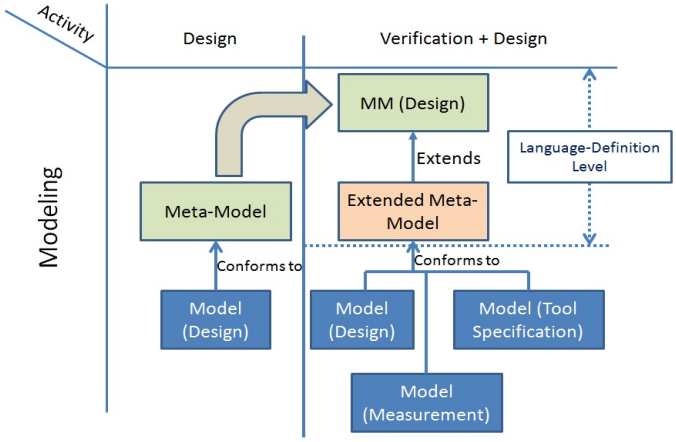

Fig. 7. Extending the Design MM to include new Verification Entities

VII. Contributions

In this section, we present our contributions in order to achieve our method shown in (Fig.5).

A. Linking Verification and Design Activities on the level of design language

In order to verify the TS earlier and at the design time, we have extended the syntax of the design language (DSML) defined in the Meta-Model (MM). The new extended MM includes the same entities that where defined for the design purpose [5], and it includes new verification entities (e.g. Measurement, Tool Specification) (Fig.6) that are connected to each other and to the design entities to provide a new syntax that can be used for an extended DSML. This extended DSML includes entities for the design and verification activities (Fig.7). The merge and linking between both design and tool oriented entities makes it possible to select the right tool for the required measurement. The need to the measurement is linked to the requirements that are represented in the softgoals (section III).

The probe entity in (Fig.6) plays a central role in the tool selection algorithm, as it is connected to both measurement and tool/simulator entities by an aggregation relationship [18]. This makes it possible to model the probe in two models: the tool specification that is related to the tool, and the measurement specification that is related to the standards and definitions of the measurements. We use the name of the probe entity from the OPNET simulator. In OPNET simulator [22], it is possible to generate output data during simulations. This

[1]We refer to the description activity of tool specifications using models by tool modeling, and the tool-specification description activity using models by specification modeling.

data can be statistics, automatic animation, and customized animations. Our general approach takes the customization of measurements into consideration, this makes it coherent with network simulators. In NS-3 simulator, we consider the trace entity as equivalent concept to the probe in our meta-model. We have been succeeded recently in generating simulation code associated with measurement configuration using the trace and log concepts [23], where we have mapped the probe concept in the proposed meta-model to the trace one in the NS-3 simulator.

B. Tool Selection Algorithm

The tool selection is done by a model transformation, we choose XPAND language [9] that is specific to generate plain-text files from models. Figure.8 presents the algorithm of the tool selection method, where the input of the algorithm is the required measurement. This measurement is identified in a previous activity where we select the needed measurements from the predefined softgoals.

We present the algorithm as : for a specific measurement $i \in Y$ (Y is the size of a set of measurements) , we iterate N times (N is the size of a set of tools), so to register which are the tools that have the probes needed (capability) for the specified measurement. The result is a set of tools that are selected by the algorithm, these tools form with other types of tools a tool-chain later in our approach.

We highlight the advantages when using our method:

1) The reusability of the model transformation, as its input can be of different types (e.g. Measurements and Certification level) (Fig.7), and rules of the transformation are defined with no hard coding (cf. Figure9). This should answer the criteria 1 in section II;
2) Our method distributes the complexity of tool selection method between two activities: Modeling, and model transformation;
3) The measurement may be previously implemented in tools, or they can be customized and then use different types of probes. Our approach takes this point into consideration and offers the ability of designing customized measurements. Our tool selection method takes this aspect into consideration. Thus, the tool is selected when it contains all of the probes that are needed by a measurement;
4) There is one loop for the measurement selection (Y times) and 3 other nested loops that are used in the model transformation template (Fig.9) and the complexity of the algorithm is computable due to its simplicity.

We compute the complexity of the proposed algorithm taking into consideration the worst case, as the number of probes is different from one tool to another and it is also different between measurements.

$Complexity = \theta(Y + N * max(PM) * max(PT))$;

θ: is the asymptotic estimation of the complexity, where the accurate result may differ due to the efficiency of the implementation of the algorithm. PM is number of probes defined for Measurements, PT is the number of probes for tools (simulators), N number of tools (simulators), and Y is the number of the measurements.

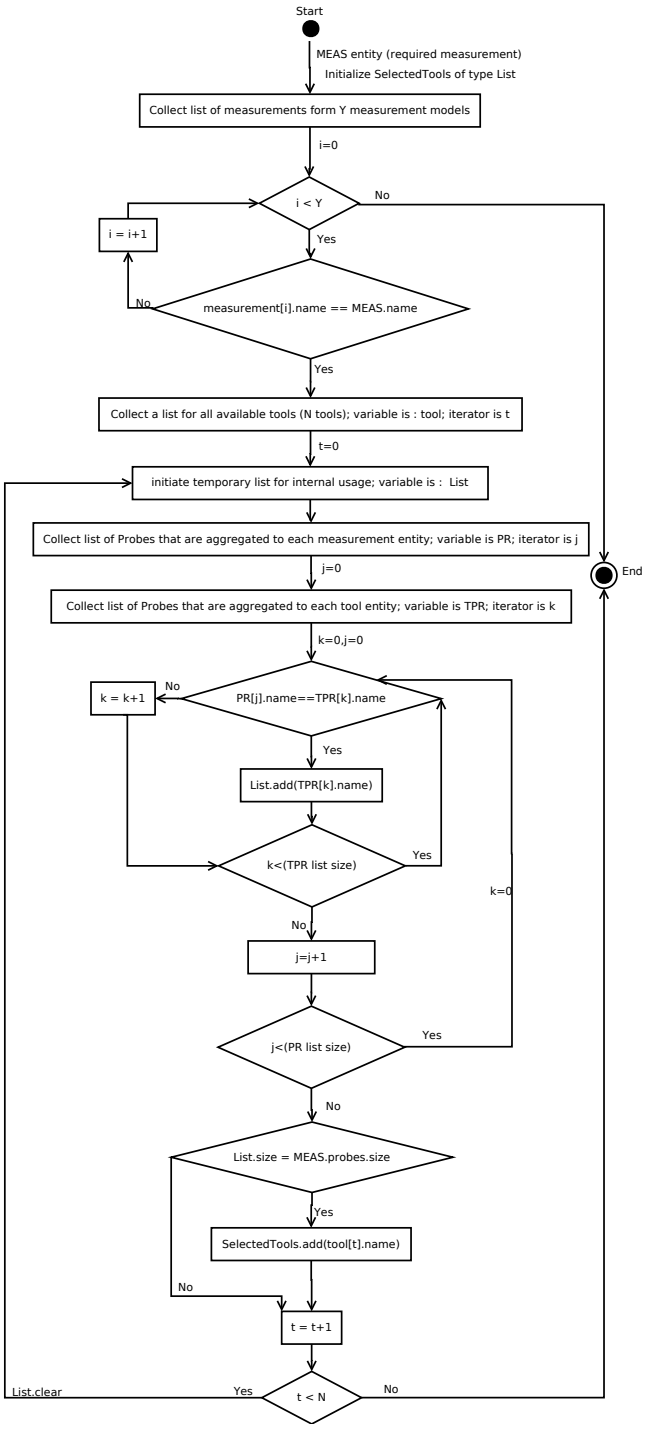

Fig. 8. The algorithm of the Tool Selection method

VIII. Video Conference Example

A. Video Conferencing Requirements

One TS may contain several tasks, and each task may trigger different types of applications. There is always a connection between the available resources of a system and the application

```
«IMPORT metamodel»
«DEFINE main FOR Model-»
«FILE  "SelectedTools.xml"-»
    «FOREACH this.eRootContainer.eAllContents.typeSelect(Measurement) AS meas ITERATOR i-»
        «IF meas.name.matches("SessionPeriod")-»
            «FOREACH this.eRootContainer.eAllContents.typeSelect(Simulator) AS tool ITERATOR X-»
            «LET {} AS list-»
                «FOREACH meas.probes AS PR-»
                    «FOREACH tool.probes AS TPR-»
                        «IF PR.name.matches(TPR.name)-»
                        «list.add(TPR.name).replaceAll(".*","").replaceAll("[ ]+","")-»
                        «ENDIF-»
                    «ENDFOREACH-»
                «ENDFOREACH-»
                «IF list.size.toString().matches(meas.probes.size.toString())-»
                «X.counter0-». Tool «tool.name-» is selected.«ENDIF-»
            «ENDLET-»
            «ENDFOREACH-»
        «ENDIF-»
    «ENDFOREACH-»
«ENDFILE-»
«ENDDEFINE»
```

Fig. 9. XPAND template using Eclipse Software

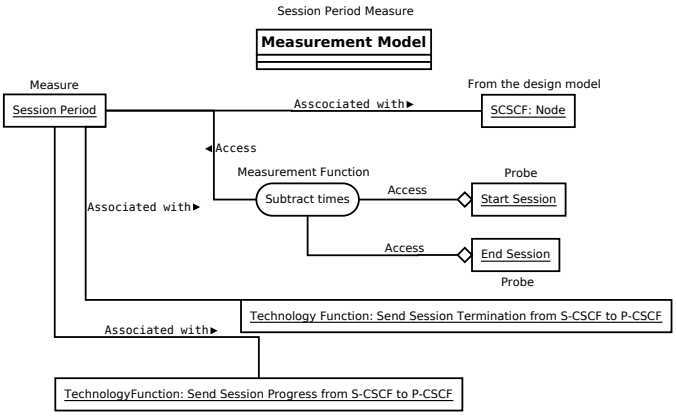

Fig. 10. SessionPeriod Measurement model

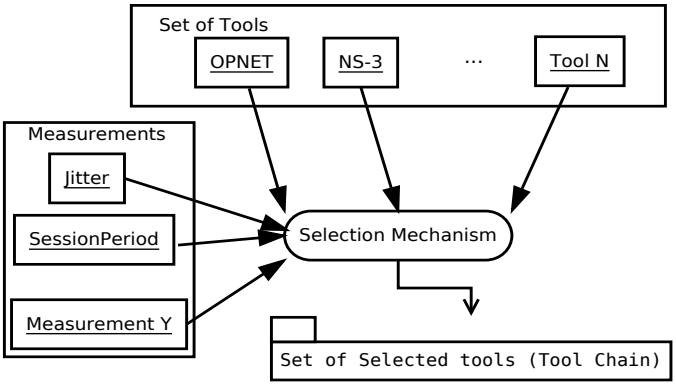

Fig. 11. Example of Tool-Selection instance as applied in the network simulation domain

QoS requirements. The System resource management makes trade offs when deciding the dedicated resources for each session of a TS. This is related to the QoS profile of the user.

In order to early estimate the performance of the TS, we link the functional and non-functional requirements to the measurements that are modeled. These measurement models are responsible to generate the measurement configuration that a tool should run. They are connected with the design entities (structural and behavioral).

The QoS of video conferencing TS are presented in [24]. The authors presents a multi-dimensional approach to analyze and manage the system resources due to the QoS requirements. These requirements are categorized in: Data Delivery Reliability, Video Related Quality, Audio Related Quality. We choose some of the parameters for our case study such as end-to-end delays and Packet Rate [10].

Such measurements are not available in all network simulators, although they can be implemented. In our approach, we intend to use tools not to develop them. Thus, we have implemented the tool selection method, so to select the proper tool due to the requirements that are specified previously in the requirements' phase (section III).

B. Measurement Modeling and Tool Specification

In this subsection, we present a customized measurement model which target is to measure the session period (Fig.10) as an example of measurements that are not already built in network simulators.

This answers the criteria 1 and 5 in (section II). This means that we add the ability of designing measurements with reuse ability. The rest of the widely-used measurements that are needed for the verification of Video Conference TS are already implemented in some network simulators (e.g. OPNET, NS-3).

C. OPNET and NS-3 case study

The measurement models are fed to the model transformation (Fig.11) that is implemented using XPAND (Fig.9), and uses the extended Meta-Model (Fig.6) as a type-definition source. The relation between the tool model and the measurement model is obvious in (Fig.6).

We add the instance model of the customized measurement (Session Period) to the NS-3 simulator properties to enforce the tool selector to choose NS-3. NS-3 accepts C++ or Python languages to configure simulation scenarios. This enables us to implement the measurement functions in a flexible way and using code generation, especially when they have mathematical operations (subtraction in the case of SessionPeriod measurement). This case study aims to illustrate our method, and the result is a list of the selected tool(s) due to the probe availability.

IX. Conclusion and Future Work

In this paper, we have proposed a method for a tool selector as generalized for different types of properties that a tool may include. This method is implemented as an activity in our general approach for detecting the errors and predicting quality violations in a telecom service design earlier before the implementation activity. Applying MDE makes it possible to find model-based solutions for the different activities and challenges in our approach. We link between the models to produce other models or codes that are inputs for another

stage or can be themselves results. The choice of tool-chains provides a solution to the variety of measurements. Our tool selection method is based on model transformation that accepts measurement and tool models as inputs and produce a list of the proper tools, where another model transformation will handle the code generation to obtain the configuration script that is needed to configure every simulator. Additionally, we have presented the model transformation rules as they guide the tool selection algorithm using iterative methods. Our method takes into account customized measurements that may rely on different types of probes. Our tool selection method decomposes the complexity of the tool selector between modeling and rule implementation activities. This reduces the complexity of the transformation rules. The reusability of our method is achieved thanks to the meta-model extension that we have proposed, and the transformation rules that are clear from hard-coding. Using XPAND model transformation makes it possible for us to integrate the results of the tool-selection with other activities in our approach. On the other side, the measurement and tool-specification modeling needs domain and modeling experience. Although models are reusable, but measurement and tool specification modeling is a task that needs high accuracy and experience, and consumes a considerable time.

For the future, we are investigating using requirements to select the set of measurements needed in the verification activity is an important part of our near future research. We are going to investigate on the model checkers to be able to check the behavioral view of the TS design. Additionally, we are investigating the integration between the different tools that forms the tool-chain. This integration should result in a set of measurements that are going to be analyzed to obtain the feedbacks that corrects the design or the requirements.

ACKNOWLEDGMENT

The authors acknowledge the support of OPNET community in providing their academic edition, which made it possible for them to use and integrate the simulation tools in their transformation chain.

REFERENCES

[1] H. Berndt, G. Peter, and W. Masaki, "Service specification concepts in tina-c," in *Proceedings of the 2nd Intl. Conf. on Intelligence in Broadband Services and Networks: Towards a Pan-European Telecommunication Service Infrastructure*. London, UK: ACM, 1994, pp. 355–366.

[2] P. Combes and B. Renard, "Service validation," *Computer Networks*, vol. 31, no. 17, pp. 1817 – 1834, 1999.

[3] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *Non-Functional Requirements in Software Engineering*, V. R. Basili, Ed. Kluwer Academic Publishers, 1999.

[4] P. M. Azuma, *ISO/IEC CD 25000.2 - Software and Systems Engineering – Software product quality requirements and evaluation (SQuaRE) - Guide to SQuaRE*, Joint Technical Committee ISO/IEC JTC 1, information technology, Software and System Engineering, ISO Std., April 2003.

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

[6] J. Simonin, Y. Le Traon, and J. M. Jezequel, "An enterprise architecture alignment measure for telecom service development," *11th IEEE International Enterprise Distributed Object Computing Conference, Proceedings*, pp. 476–483, 2007.

[7] G. A. Moreno and P. Merson, "Model-driven performance analysis," in *Conference on the Quality of Software Architectures (QoSA 2008)*, ser. LNCS, F. P. S. Becker and R. Reussner, Eds. Karlsruhe, Germany: Springer, October 14-17 2008, pp. 135–151.

[8] A. Achilleos, K. Yang, N. Georgalas, and M. Azmoodech, "Pervasive service creation using a model driven petri net based approach," in *Wireless Communications and Mobile Computing Conference. IWCMC '08.*, Aug. 2008, pp. 309 –314.

[9] Eclipse modeling. Eclipse. http://www.eclipse.org/modeling ; Last visited on 15-February-2013.

[10] 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. Springer Berlin Heidelberg, August 2012, vol. 7479, pp. 100–111.

[11] 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)*, Seville, Spain, 2011, pp. 23–29.

[12] F. Arbab, S. Meng, and Y.-J. Moon, "Reo2mc: a tool chain for performance analysis of coordination models," in *ESEC/FSE '09 Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, vol. I, Amsterdam, The Netherlands, August 24–28 2009, pp. 287–288.

[13] D. Ivanov, M. Orlić, C. Seceleanu, and A. Vulgarakis, "A set of integrated tools for behavioral modeling and analysis of embedded systems," in *ASE '10 the proceedings of the IEEE/ACM international conference on Automated software engineering*, 2010.

[14] B. Berthomieu, J.-P. Bodeveix, S. D. Zilio, P.Dissaux, M. Filali, P. Gaufillet, S. Heim, and F. Vernadat, "Formal verification of AADL models with FIACRE and TINA," in *Embedded Real Time Software and Systems (ERTS) 2010*. CNRS and Airbus, 2010.

[15] P. Dissaux and F. Singhoff, "Stood and cheddar: AADL as a pivot language for analysing performances of real time architectures," in *Proceedings of the European Real Time System conference. Toulouse, France*, 2008.

[16] 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.

[17] L. Chung and J. do Prado Leite, "On non-functional requirements in software engineering," in *Conceptual Modeling: Foundations and Applications*, ser. Lecture Notes in Computer Science, A. Borgida, V. Chaudhri, P. Giorgini, and E. Yu, Eds. Springer Berlin / Heidelberg, 2009, vol. 5600, pp. 363–379.

[18] The Open Group, *ArchiMate 1.0 Specification*, The Open Group Std., 2009.

[19] 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.

[20] V. Chiprianov, Y. Kermarrec, and S. Rouvrais, "Meta-tools for Software Language Engineering: A Flexible Collaborative Modeling Language for Efficient Telecommunications Service Design," in *ICSE Workshop on Flexible Modeling Tools*, 2010.

[21] M. van Amstel, "The right tool for the right job: Assessing model transformation quality," in *Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual*, july 2010, pp. 69 –74.

[22] X. Chang, "Network simulations with OPNET," in *Winter Simulation Conference Proceedings*, vol. 1, 1999, pp. 307 –314 vol.1.

[23] *ns-3 Manual-Release ns-3-dev*, November 30 2012. [Online]. Available: www.nsnam.org/docs/manual/ns-3-manual.pdf

[24] C. Lee, J. Lehoczky, D. Siewiorek, R. Rajkumar, and J. Hansen, "A scalable solution to the multi-resource qos problem," in *Real-Time Systems Symposium, 1999. Proceedings. The 20th IEEE*, 1999, pp. 315 –326.