Extending Telecom Service Design Activities for Early Verification

Iyas Alloush\textsuperscript{1,2}

Supervisor of the thesis: A/Prof.Siegfried Rouvrais\textsuperscript{1,3}
Director of the thesis: Prof. Yvon Kermarrec\textsuperscript{1,2}

\textsuperscript{1} Telecom Bretagne, Institut Mines-Telecom, Université européenne de Bretagne
Technopole Brest Iroise, CS 83818 29238, Brest Cedex 3, France
\textsuperscript{2} UMR CNRS 6285 Lab-STICC
\textsuperscript{3} IRISA

1 Introduction

The production of Telecommunication Services (TS)s is a long term process and consumes many resources. A mistake in the TS design may result in expensive consequences to the rest of the TS service life cycle activities [5]. It is difficult and cost expensive to fix errors and flaws in the design after the implementation activity, especially if there are hardware elements to be installed. Therefore, we will take into consideration two points: the TS characteristics, and the TS creation activity.

The TS have several characteristics that make them different from other software systems. The platforms that a TS relies on are large scaled \textsuperscript{13} and used by a large number of customers. A TS may rely on software and hardware elements to perform its tasks. Traditionally, a TS is composed of numerous number of applications that interact with each other, this results in a complex TS design [6].

The contribution of this thesis is in the scope of TS creation activity (TSC) [2], and relies on a recent dissertation [7]. We address the early verification activity right after the design one, and before the implementation phase. The aim of the verification activity is to “check that the software meets its stated functional and non-functional requirements” [14]. We consider a very narrow part of the Performance Non-Functional Requirements (PNFRs) [8], where there are space (e.g. memory utilization) and time (e.g. throughput and response time) PNFRs. The Service Creation Environment (SCE) [3] is a collection of software tools that rely on reused infrastructure which is used according to the service development methodology, to assist the service developer(s) by automating and simplifying the Service Creation process. We make an early verification of TSs before the implementation using automated methods to hide complexity from the service designer and reducing code implementation errors.

We face the following challenges: (1) To acquire the customer satisfaction by meeting the requirements of the QoS of a TS (e.g. Video Conference QoS requirements [11]), and few PNFRs that meet both the service and network providers (e.g. CPU and Memory utilization); (2) To reduce the time to market that should be as short as possible without affecting the qualities; (3) Improving the cost efficiency.
Our main objective is to assist the different stakeholders (Service and Network Providers) that are involved in the TS specification activity considering the different actors (Service Designers, Service Developers, and End-User), and taking into account three important factors that are mentioned previously. The detection of a design error or flaw after the implementation of software and hardware elements of a TS may result in expensive consequences related to the upper-mentioned competition factors.

2 Research Questions

According to our research advance, we have branched the problematic to the following research questions: *(RQ1)* How to detect the TS design errors and flaws earlier and before the implementation phase?; *(RQ2)* How to correct the design and improve the qualities accordingly?

The first research question is divided to 4 branches: *(RQ1.1)* How to Link between softgoals [8], design, verification activities and to identify the measurement(s) that are related to a specific requirement?; *(RQ1.2)* How to select automatically the proper measurement(s) accordingly?; *(RQ1.3)* How to select the proper network simulator that can provide the specific measurement(s)?; *(RQ1.4)* How to link the modeling technical space to the simulation one in the scope of TS verification activity?

Regarding to the second research question RQ2, we raise the following questions: *(RQ2.1)* How to analyze the measurements that are resulted from the simulation activity?; *(RQ2.2)* How to decide if a modification on the design is needed to be taken automatically?

3 Approach

Relying on Model Driven Engineering (MDE) and Model Driven Analysis (MDA) [12], we propose to verify the TS through its design models at the TSC activity.

Model transformation (MT) is a key concept in MDE to transform automatically a source model to another one. Eclipse IDE supports different MT languages such as XPAND, as well as modeling tools through the Eclipse Modeling Framework (EMF).

MDA is an approach that aims to verify if the input design model meets the quality requirements [12]. Our approach as presented in (Fig. 1) is coherent with the MDA approach in [12].
4 Methodology

We propose to divide the verification activity into 4 main sub-activities (Fig. 2): (1) **Linking activity**: to link between the design architecture, measurement(s), and requirement models, thus prepare inputs for the next activity (the model adaptation activity); (2) **Model Adaptation Activity**: to link between the design and the tool technical spaces in general, and network simulators in specific; (3) **Measurement Analysis**: to make the necessary analysis based on an analytic theory (e.g. Queuing Theory) taking the measurements that are obtained from the simulation or model checking tools (tool-chain components) as an input; (4) **Feedback Activity**: to correct the design and requirement models according to the results of the previous analysis activity.

5 Contributions

In this section, we present our contributions to answer the previously mentioned research questions:

1. We have extended the design Meta-Model (MM) (abstract syntax) in [7] to include verification-related elements: (1) Modeling the measurement element to contain different probes and an aggregation function called (measurement function) [published in P3, and modified in P5]. This contributes to the RQ1 with all of its branches; (2) Associating the measurement elements with the design ones (behavioral and structural) together through the probes. This association forms a transversal alignment between the measurements and ArchiMate [15] (Fig. 3) thanks to the (one-to-many) aggregation relationship between probes and measurements in the MM [submitted in P5]. This provides an approach that contributes to RQ1.1 and establishes a link between the softgoals and measurements through the business layer of ArchiMate. This will contribute to RQ1.2; (3) Adding Open Systems Interconnection (OSI) elements that are related to the protocols in the technology layer of ArchiMate (e.g. Protocols for layers 3 and 4). [published in P3]. This contributes to RQ1.4.

2. Mapping directly between the design and the network simulation technical spaces thanks to the new elements that are added to the MM and to XPAND code generation language [published on P3 and submitted in P5]. This contributes to the RQ1.4;
3. We have proposed a Tool-Selection method to select the proper tool according to its capabilities, where our interest till now in our research is in the measurements [published in P4], this contributes to the RQ1.3.

6 Related Work

In this section, we provide a rapid view to some of the related work according to the following aspects: (1) Managing design and verification complexity; (2) Involving different stakeholders in the design and early verification activities; (3) Taking the hardware elements of the TS design into consideration; (4) The ability to use different types of verification tools.

The authors in [1] tackle the context-awareness at the application level relying on MDE approach, as it facilitates the design and implementation of pervasive services, this also helps in managing the design complexity. They propose a context-aware modeling framework. The end-user viewpoint is taken into account. Their approach considers models of high abstract level and the behavioral concepts of the service specifications. They uses code generations after several mediation model transformations to validate the pervasive service. Although we both rely on MDE and Meta-Modeling, but there is a major difference between our approaches, that is we take multi-stakeholder viewpoints into consideration, and perform early verification using simulation approach and relying on tool chain formalism from the design to the correction activities.

In [10], the authors propose REMES tool chain that provides the system designer with a series of tools from the design editor to the system testing and verification ones relying on model transformation techniques. This approach is too close to ours, while it considers the behavioral concepts only. Thus, this approach can’t represent TS nature precisely as it lacks to the hardware representation. The code that is generated for REMES simulator describes clocks, model transitions, and variable changes, which are all related to model checking and behavioral testing activities, while no mention to execution platforms that may contain hardware elements such as servers.

The authors in [4] work in the domain of coordination modeling, where they rely on Constraint Automata (CA) to analyze the coordination models. One can realize that they consider the behavioral concepts only in their design and evaluation activities, thus their approach relies on model checkers (PRISM in their case study). Relying on MDE, they propose a tool chain to perform performance analysis of coordination models. They rely on Eclipse IDE in their implementations of the design and evaluation tools, where they use Reo plug-in. Although our both approaches rely on tool-chains, but the major difference is that we represent hardware elements, use simulators, and include a tool selection method in our proposal that makes it possible to use different simulators.

In [16], the authors propose an environment (SimStudio) for modeling and simulation based on DEVS formalism (Discrete EVent Systems Specification). Their proposal offers a tool chain for the modeling, simulation, and analysis activities. Their proposal seems very close to ours, except that we propose tool
selection method, and we propose an approach to link between goals and early verification activities through our recent proposal in P5. Additionally, we have proposed a direct link between the design models and the network simulators such as OPNET and NS-3 [P1, P2, P3]. Thus we consider both of the domain specificity of the underlying platform PSM and the higher level models PIMs.

7 Conclusion and Future Work

In this report, we have proposed the abstract syntax of Domain Specific Early Verification Language (DSEVL). It is an extension to a previously defined DSML [7], and relies on ArchiMate language as well, where ArchiMate shares the design model between the different viewpoints. We have proposed a tool-chain approach that covers the different activities from design to verification by simulation where we extend a previous work on the design activity [7]. A test-bed is implemented relying on MDE discipline and network simulators. We have contributed to a part of our research questions and looking forward to do the same to the rest. Relying on this tool-chain, we stand focused on the objective of our thesis as we define a work flow which is composed of tools from different domains, specifications, and technical spaces.

Our work takes into consideration the different viewpoints of enterprise stakeholders thanks to EA framework [9]. Relying on MDE and MDA helps in managing the complexity of both activities: design and early verification. These model transformations are useful as they reduce the time of the verification implementation and the possible errors that can be made manually. We bring the measurement concept from the operation activities to the design one relying on the benefits of using simulators, where no implementation cost is to be considered.

On the other side, our approach considers different domains and technical spaces, which needs domain specific experts to model and implement the composing tools. Besides, implementing the test-bed including different simulators is a time consuming activity and needs also domain experience.

For the future, we intend to define the concrete syntax of the early verification language that extends the Domain Specific Modeling Language (DSML) [7]. We intend also to investigate in the problem that has faced us when aggregating the data collected from the different layers of ArchiMate (e.g. Business, and Technology) [problem declared in P5]. Additionally, we continue to investigate the rest of the research questions.

8 Publications

and Enterprise Architecture for Early Verification of Telecom Service Design, 2013. [Submitted]

References

15. The Open Group: ArchiMate 1.0 Specification (2009)