Simulation Modelling Practice and Theory 66 (2016) 243–259

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Simulating simulation-agnostic SysML models for enterprise information systems via DEVS

Anargyros Tsadimas*, George-Dimitrios Kapos, Vassilis Dalakas, Mara Nikolaidou, Dimosthenis Anagnostopoulos

Department of Informatics and Telematics, Harokopio University of Athens, 70 El. Venizelou St, Kallithea, 17671 Athens, Greece

ARTICLE INFO

Article history:
Received 11 January 2016
Revised 1 April 2016
Accepted 1 April 2016

Keywords:
Automated simulation code generation
Requirements verification
Model-based system design
SysML
DEVS
Non-functional requirements
Model transformations
MDA
EIS

ABSTRACT

Systems Modeling Language (SysML) is used as the modeling infrastructure in systems engineering, especially for complex systems design, independently of the system domain. Simulation is a common method to perform system model verification, during the systems development process. However, simulation code generation and execution is not integrated within the system design activity, as it is facilitated by SysML. It is either conducted as an external activity, after system design, or it affects the system design environment and practices, according to specific simulators requirements.

This paper presents how existing, simulation-agnostic SysML models from the domain of Enterprise Information System (EISs), can be transformed to executable simulation code and in addition how the simulation results can be incorporated into the source SysML model through the exploitation of Model Driven Architecture (MDA) principles and techniques. To this end, several tools and technologies are utilized, while the verification process is triggered and finalized via the system modeling environment. Adoption of MDA provides a solid, high-level infrastructure and tool availability to the proposed approach.

1. Introduction

Model-based Systems Design (MBSD) deals with the definition and exploitation of system models during most of the system development activities. Modeling Language (SysML) [1] has been proposed by the Object Management Group (OMG) to serve as a common modeling language for all these activities. It is a general purpose language, based on a variety of modeling elements grouped in different diagrams to support complex system models. SysML is a well-established language for complex system modeling and is often used in MBSD. It facilitates the description of both system structure and behavior, while design requirements may also be described in an abstract level [1]. In the work presented in this paper, emphasis is given on requirements, related to system performance, that may be verified against quantitative estimations of the system behavior [2]. A common way for deriving such estimations is simulation. Thus, there is currently strong interest in generating simulation models from SysML models [3,4].

Today, system designers utilize domain-specific profiles [5–8] defined for SysML modeling tools combined with external simulation tools to explore the performance of alternative system designs and verify corresponding requirements [9]. Usually the system designer should be familiar with both SysML models and the simulation environment and feel confident

* Corresponding author. Tel.: +30 210 9549418; fax: +30 210 9549 401.
E-mail addresses: tsadimas@hua.gr (A. Tsadimas), gdkapos@hua.gr (G.-D. Kapos), vdalakas@hua.gr (V. Dalakas), mara@hua.gr (M. Nikolaidou), dimosthe@hua.gr (D. Anagnostopoulos).

http://dx.doi.org/10.1016/j.simpat.2016.04.001
S1569-190X(16)30025-9/

in working with corresponding tools. While system design models are defined in SysML modeling tools, requirement verification and the evaluation of the proposed solution is usually performed within the simulation environment. In order to simplify the overall process, system designers should be provided with a single system model for both system design and requirement verification, while simulation model generation should be fully-automated and transparent to them. Furthermore, SysML system models should be defined, independently of the methods and tools adopted for simulation. Such a task still remains a challenge [10].

The work presented in this paper addresses these issues, by enhancing the process of SysML models evaluation and simulation code generation, without requiring the integration of simulator-specific properties in the profiles used during system design. These enhancements simplify the evaluation process, allowing the system designer to focus on the examination of the unverified requirements and, consequently, the detection of the necessary design solution re-adjustments [6]. Model Driven Architecture (MDA) [11] principles and technologies have been adopted to automate simulation code generation and integrate simulation results into the SysML modeling environment. To explore the proposed approach, we study its applicability in a specific domain, namely Enterprise Information Systems Design, using a specific simulation framework, namely Discrete Event System Specification (DEVS) [12].

Our approach provides simulation capabilities for SysML system models, without requiring the use of a simulator specific SysML profile. This renders the approach simulation-agnostic in its conception, establishment and influence to the system designer. Enterprise Information System Design is supported by the EIS SysML profile, described in [6]. No simulator-specific properties are defined in the profile. On the contrary, it focuses on different views for EIS architecture, while it enables requirement verification and the definition of evaluation scenarios. The EIS profile is examined and utilized to develop a framework that extracts information, necessary to derive executable DEVS simulation models, based on DEVS library components. DEVS was chosen, since (a) system structure is defined in both SysML and DEVS in a similar fashion, making the transformation conventional, and (b) existing tools facilitated standardized DEVS meta-model representation and enhanced simulation code generation process [4]. SysML-to-simulation model transformation needs to be defined only once for the pair of the Enterprise Information System (EIS) domain-specific SysML profile and the DEVS simulation environment, ensuring a seamless integration of SysML modeling environments and DEVS simulators for the domain of EIS. While it is applied, simulation integration is transparent to the system designer that interacts only with the SysML EIS models within a SysML modeling environment. This way, no additional simulation-related restrictions are imposed to system designers. The proposed approach for MBSD may be easily utilized in numerous domain system models, once an appropriate simulator, for the specific domain, has been chosen and the corresponding model transformation has been developed.

The rest of the paper is organized as follows: In Section 2, related work is discussed, describing relevant approaches. The pre-existing SysML profile for the domain of EIS is presented in Section 3, where specific issues and challenges that need to be addressed are also outlined. Section 4, presents a requirements verification scheme that aims at addressing these issues. It is based on MDA principles and techniques and exploits the DEVS simulation framework, in order to provide estimations about the performance of system model configurations. Section 5 provides a practical perspective of the design and evaluation process and, finally, conclusions and emerging issues from this implementation reside in Section 6.

2. Related work

SysML supports a variety of diagrams describing system composition and states, necessary to perform simulation, which are utilized by different approaches [13,14]. In most cases, SysML models defined within a modeling tool are exported in XML Metadata Interchange (XMI) format and, consequently, transformed into simulator specific models to be forwarded to the simulation environment. Depending on the nature and specific characteristics of the systems under study, there is a diversity of ways proposed to simulate SysML models, utilizing different diagrams. The authors have recently presented a comprehensive understanding of the similarities and differences of existing approaches targeting simulation code generation from SysML models in [10], where current challenges in fully automating simulation of SysML models are also identified. The most well-known of them are briefly discussed in the following.

The SysML4Modelica profile [15], endorsed by the OMG, enables the transformation of SysML models to executable Modelica simulation code. The Query/View/Transformation (QVT) standard set of languages is used for the transformation of SysML models to executable Modelica models. Since SysML profiles are based on the Unified Modeling Language (UML) extension mechanism, they can be imported in any standard UML modeling tool, such as Rational Modeler [16] or MagicDraw [17], enabling the utilization of simulation tools. In [18], focus is given on embedded systems. In the proposed profile, the SysML requirement entity is extended with testable characteristics. Testable requirements are associated to conditions under which the requirement is verified with the use of experiments or test cases. Verification conditions are defined as part of a test case, which in turn may be simulated using Modelica simulation language in external simulators to ensure that a design alternative satisfies related requirements [18]. To embed simulation capabilities within SysML, the ModelicaML profile is used. Verification conditions associated to testable requirements are also defined in ModelicaML [19], while requirement verification is performed in an external Modelica tool (MathModelica), through visual diagrams created during simulation. This effort also does not return any feedback back to the SysML modeling tool. The key limitation of this approach is that it is tailor-made to the Modelica simulation environment, in order to describe the requirements and the verification process together.