Obtaining a Set of Recommendations for Evolving Executable Languages towards Systems-of-Systems Architecture Design*

Bruno G. A. Lebtag (Student)[1], **Valdemar V. Graciano Neto (Supervisor)**[1]

[1] GoIn-Sight: **Go**iás **In**formation **S**ystems and Software eng**i**neerin**g** Researc**h** **T**eam
Mestrado Acadêmico em Ciência da Computação
Programa de Pós-Graduação em Ciência da Computação
Instituto de Informática – Universidade Federal de Goiás (UFG)
Caixa Postal 131 – CEP 74001-970 – Goiânia – GO – Brazil
Admission: 03/2019 – Qualification: 06/2020 – Expected period of defense: 02/2021

{brunogabriel@inf.ufg.br, valdemarneto@ufg.br}

Abstract. *Systems of Systems (SoS) are complex systems composed of managerially and operationally independent constituent systems (CS). Smart cities are examples of SoS. However, these types of systems impose challenges to traditional software architecture such as highly evolutionary architecture due to the evolution of individual CS and emergent behaviors that results from the interoperability of CS. Executable Models (ExM) are a class of models that can be executed and that can assist on architectural design of SoS. By using them, architects can predict the SoS structure and behavior by visualizing and simulating the SoS still at design-time. On the other hand, as any other emergent technologies, it suffers with absent scientific evidences of its benefits, mainly in industrial context. The main contribution of this master thesis project intends to be the presentation of evidences about the use of ExM to solve problems in the SoS software architecture design. For achieving such purpose, this project is structured in well-defined steps: (i) a systematic mapping study, (ii) elaboration of a conceptual map derived from the mapping study, (iii) a survey to obtain perceptions from software engineering professionals on the use of ExM in the context of engineering simple systems and (iv) a new survey expanding the previous study and to obtain perceptions from software engineering professionals on the use of ExM in the context of architectural design of SoS. Preliminary results reveal that ExM have been reported in diverse SoS domains, offering a dynamic and interactive view to the SoS. We also found that software engineering professionals see ExM as suitable solution for dealing with complex and critical systems as SoS.*

Keywords: *Systems-of-Systems, Executable Models, Software Architecture.*

Related events: *SBCARS, SBES*

*This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

1. Introduction

Systems-of-Systems (SoS[1]) are a type of *complex systems* [Northrop et al. 2006, Maier 1999] that represent a new challenge to system architecture (SA) [Jamshidi 2008]. SoS are composed of managerial and operational sub-systems known as *constituent systems*. Smart cities are an instance of SoS [Manoj Kumar et al. 2018]. In a smart city, its architecture is constantly changing and evolving. Its citizens are frequently arriving and leaving the city while bringing together their appliances capable of communicating via the internet (so called smart gadgets) which are capable of contributing and affect the smart city architectural configuration. Executable technologies, such as models@runtime [Gu et al. 2018], executable UML [Ciccozzi et al. 2018] and simulation [Gray and Rumpe 2016] which are known as Executable models (ExM) allow software architects to design and evaluate the architecture and the emerging behaviors of their constituent systems [Levis and Wagenhals 2000].

ExM allow the architect to (i) visualize systems structure and behavior still at design-time, (ii) predict and evaluate the consequences of architectural changes (iii) model and observe the interactions of constituent systems and (iii) propagate such changes in the source model to preserve the documentation updated. However, ExM can still be considered an emergent technology [Wang and Dagli 2011] since it has not even been fully explored beyond academic context [Ciccozzi et al. 2018].

The main goal of this study is to contribute to the research area of using ExM for architectural design of SoS. Our specific goals are: (i) providing a mapping of the state of art and to produce a conceptual map of the main concepts involved in the area, (ii) evaluating existing ExM notations (using survey and experiment) about their usability and suitability to support SoS architectural design from a technical and professional perspective, and (iii) compiling the results in a list with benefits and opportunities perceived by professionals to foster ExM evolution to cope with SoS demands. Preliminary results reveal that (i) architectural evaluation and synthesis (via simulation) is very important for SoS architectural design, (ii) software engineering professionals consider ExM suitable for complex and critical applications while simultaneously serving as a communication document.

This paper is organized as follows: Section 2 presents the paper background; Section 3 describes the scientific methodology and evaluation plan; Section 4 presents the preliminary results and achievements; Section 5 brings the expected contributions; finally, Section 6 concludes the paper with the related works.

2. Background

SoS is an emergent research area. As any other research area, it also suffers from divergent definitions and perspectives. One of the most used definitions was proposed by [Maier 1999]. He defines SoS in terms of five main characteristics: (i) Operational Independence (OI) of CS, i.e., each individual CS is independent and able to operate when disassembled from the SoS, preserving its autonmy; (ii) Managerial Independence (MI) of constituents, since CS present a degree of self-governance and are managed and owned

[1]Herein, SoS acronym will be interchangeably used to express both singular and plural forms: System-of-Systems and Systems-of-Systems.

by different organizations and entities; (iii) Distribution (D), which indicates that CS usually operate distributed and through interoperability links; (iv) Evolutionary Development (ED), as the entire SoS evolves as a results of individual CS and interoperability links evolution; and lastly (v) Emergent Behavior (EB), which is the global behaviors achieved through CS interoperability. As a result of OI, the SoS architecture is dynamic or evolutionary [Oquendo 2016], i.e., constituents addition, replacement, removal or the entire SoS dissolution and rearrangement constantly leads the SoS to novel architectural arrangements known as coalitions [Manzano et al. 2020].

These five characteristics represent challenges to traditional architecture practices [Jamshidi 2008]. Distribution forces architects to pay special attention to interoperability quality attribute. ED also poses as complicating factors for the architecture, once traditional architectures are not so dynamic and current notations are not prepared to capture such dynamism, which pressures architects to search for new alternatives, such as ExM and Adaptive Systems [Jamshidi 2008, Gu et al. 2018]. EB is another problem that architects must face. They can represent new opportunities to be explored as well as threats to the security and stability of the whole system. Because SoS are usually explored in critical domains such as military domain [Jamshidi 2008], researchers are constantly investigating new ways to deal with them and looking for alternatives to manage unknown and uncertain conditions. Thus, architectural evaluation plays a major role in SoS development.

Architectures should then be evaluated. When dealing with SoS, which usually has critical nature, the architecture importance becomes evident as errors could bring serious consequences. Therefore, it is fundamental to spend time and efforts to develop the architecture and to evaluate it. However, traditional approaches struggles to deal with complexity imposed by CS independence, ED and EB. On one hand, visual models such as UML diagrams currently face a decline in its adoption by the industry. Nevertheless, they would still struggle to represent a complex and diverse system, composed of many constituents. SoS pushes graphic notation to its limit of usefulness to represent so many information as it becomes hard to maintain various models for each constituent system and to obtain some information which is scattered across different models. On the other hand, mathematical/formal models require professionals to have mathematical and formal background, an ability that can be difficult to find. Another problem shared by both approaches is maintainability. Visual and mathematical models requires to be constantly updated as the SoS evolves. In this sense, ExM can represent an advantage to traditional approaches.

ExM are models that can be executed [Dahmann et al. 2017]. Thus, they can be used directly as CS or as prototypes [Dahmann et al. 2017]. Models@Runtime [Gu et al. 2018], Executable UML [Ciccozzi et al. 2018] and Simulation Models are examples of ExM. They offer the same benefits of Visual models as they are usually accompanied by a visual representation but with the extra benefit of being executable. Architect can use them to simulate the final SoS and the interactions among components which allow them to predict and explore structure and behaviors still at design-time, thus they can use ExM to predict EB in the SoS [Jamshidi 2008, Dahmann et al. 2017]. The ED is also simplified as architects can directly modify the models and changes are automatically propagated to the underlying documentation. Those benefits are essential to SoS archi-

tecture evaluation as they can help to cope with ED and EB problems. However, ExM is also an emergent technology [Wang and Dagli 2011]. There are many opportunities yet to be explored and improved. Industry software engineering professionals can greatly contribute to improvements based on their daily experience with software development. This study aims to reduce this gap and to further consolidate the ExM research area. The next section details the methodology elaborated to achieve the study goals.

3. Scientific Methodology and Evaluation Plan

The initial hypotheses motivating this study were *"Current ExM notations are not adequate to represent SoS architecture from an industry practitioners point of view"* and *"Some ExM notations are more suitable to evaluate some SoS characteristics"*. To evaluate both hypotheses, we followed the scientific methodology presented by [Neto et al. 2010] and presented below.

Ad-hoc Literature Review: We initially conducted an ad-hoc literature review in order to obtain the main concepts involved in the area of ExM, SoS and SA. we identified: (i) the main authors of each area; (ii) important concepts and definitions; and (iii) related work.

Systematic Mapping Study (SMS): From the previous step, we identified absence of a SMS in the use of ExM for SoS architectural evaluation. A SMS serves as an overview map of current researches in area. Therefore, we performed a systematic mapping study on six different search bases to investigate how researchers are exploring ExM for SoS architectural design. From the initial results, we concluded that the use of ExM for SoS context is a gap to be further explored, particularly in the industrial context and notations suitability for evaluating SoS characteristics.

Survey with software engineering professionals: We investigated the perceptions and opinions of software engineering professionals after experiencing two small problems modeled using ExM. In this study, we exposed participants to two demonstrative simple systems in order to obtained from this experience suggestions of improvements for ExM. The intention is to obtain those perceptions and after progress to a more elaborated scenario, as SoS.

Conceptual Map: Another result we obtained from the SMS was the the lack of consensus on definitions in ExM community. As this is an important aspect for any research area, we also decided to contribute to this problem. Therefore, we developed a conceptual map with the main concepts in order to increase the standardization of ExM terminologies while also presenting a discussion about the conflicting definitions.

Survey with software engineering professionals in the architectural design of SoS: Lastly, we intend to expand the previous study to evaluate the ability of software engineering professionals to evaluate the SoS architectural design. In the previous study, we exposed participants to simple systems in order to identify problems and opinions in this level, now we further expand the study by exposing participants to evaluate different SoS architecture while collecting their perceptions and opinions.

4. Preliminary Results and Achievements

This master research project is currently half the way to reach its conclusion. The qualification exam was performed and the student was approved on June 4th, 2020.

Now, we present the preliminary results and current status of the each individual study developed in this master research project:

SMS: This study was conducted from August, 2019 through March 2020. We retrieved 196 studies and included 35 studies. Some preliminary results were the importance of architectural evaluation via simulation using mainly discrete event formalism (Colored Petri nets — CPN and Discrete Event System Specification — DEVS) and Agent-based formalism (ExtendSim formalism).
Survey with software engineering professionals: This study was conducted from, September 2019 through June 2020. We collected 58 answers from software engineering professionals around the world. Some preliminary results were the importance of improving visual notation and the necessity of specialized training for engineering professionals to use ExM.
Conceptual Map: During the SMS conduction, we collected important concepts and definitions used along the studies and we also observed conflicting definitions. Based on this results, we elaborated a conceptual map condensing those findings and presented a discussion about the conflicting definitions (e.g. ExM definition itself).

We will initiate the development of the protocol of last survey and the confection of its SoS architecture models and questionnaire during August and beginning of September, 2020. After the confection process, we will perform a pilot survey to calibrate the questionnaire and to identify any problems. Lastly, we will initiate the survey invitation by the end of September. The results will be analysed and submitted to a journal yet to be chosen. The thesis development will initiate by the end of year and the expected period of defense will be the first semester of 2021.

5. Expected Contributions

We expect to contribute to the research area with results from:

- A SMS, which provides the current state of art in researches of ExM in the context of SoS architecture design;
- A survey with software engineering professionals in the context of engineering simple systems. We present a list of advantages, disadvantages and opportunities of improvements;
- The elaboration of a conceptual map that lists the main concepts and definitions needed by the research area;
- A new survey expanding the previous study but in the context of architectural design of SoS. We also present a list of advantages, disadvantages and opportunities of improvements;

The contribution is readily a theoretical one to underpin novel advancements on ExM languages for SoS in the future.

6. Related Work

[Hojaji et al. 2019] presents a systematic mapping study regarding model execution trace. Model execution trace as defined in the study is a mechanism to capture enough relevant information related to the ExM during its execution for later analysis. The study also presents some definitions for executable modeling language and ExM. The first notable

difference from our study is scope. While [Hojaji et al. 2019] aims to characterize the use of model tracing for executable models, we are aiming for characterize the use of ExM for SoS architectural evaluation. The resulting contributions are different because the areas of interest are also different (model tracing and SoS architectural evaluation). The scope of the study is also different. While our study aims to explore ExM in the scope of SoS architecture, [Hojaji et al. 2019] has no specific scope. It is also important to notice that this study is recent and thus it is sign of the relevance of ExM.

In [Guessi et al. 2015], the authors presented a systematic literature review that investigates the notations being used to describe SoS. Although some presented notations can be executed such as UML, the study does not focus exclusively in executable notations as we do. On the other hand, In [Guessi et al. 2015], the authors compared some notations against a known framework for ADL. For our project, we selected a different approach for evaluating a single notation using a survey with software engineering professionals to obtain evidences on desirable features for executable ADL for SoS.

[Hlupic 2002] investigate the use of simulation through a survey with academic and industrial users. The study aims to identify the most used type of simulation software, common application areas among other goals. The survey aimed to identify software engineering professionals perspective through the use of toy examples followed by some questions related to the presented models. We want to collect their opinion regarding their use and any opportunity of improvement they may envision.

7. Final Remarks

This paper presented preliminary results of an ongoing master's research project. We found that research about the use of ExM for SoS architecture offers little scientific evidences of their benefits and there is a scarcity of studies exploring it in a industrial context. This master research project aims to contribute to the SoS and ExM research areas by providing a (i) SMS on ExM for SoS architecture design (ii) conceptual map with the main concepts of ExM, (iii) survey and controlled experiment with software engineering professionals to evaluate the suitability of current notations.

References

Ciccozzi, F., Malavolta, I., and Selic, B. (2018). Execution of UML models: a systematic review of research and practice. *Software Systems Modeling*, pages 1–48.

Dahmann, J., Markina-Khusid, A., Doren, A., Wheeler, T., Cotter, M., and Kelley, M. (2017). SysML executable systems of system architecture definition: A working example. pages 1–6.

Gray, J. and Rumpe, B. (2016). Models in simulation. *Software & Systems Modeling*, 15(3):605–607.

Gu, T., Lu, M., and Li, L. (2018). Runtime models for analysing and evaluating quality attributes of self-adaptive software: A survey. In *2018 12th International Conference on Reliability, Maintainability, and Safety (ICRMS)*, pages 52–61.

Guessi, M., Cavalcante, E., and Oliveira, L. B. R. (2015). Characterizing architecture description languages for software-intensive systems-of-systems. In *2015 IEEE/ACM*

3rd International Workshop on Software Engineering for Systems-of-Systems, pages 12–18.

Guessi, M., Neto, V. V. G., Bianchi, T., Felizardo, K. R., Oquendo, F., and Nakagawa, E. Y. (2015). A systematic literature review on the description of software architectures for systems of systems. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, page 1433–1440, New York, NY, USA. Association for Computing Machinery.

Hlupic, V. (2002). Simulation software: An operational research society survey of academic and industrial users. pages 1676–1683.

Hojaji, F., Mayerhofer, T., Zamani, B., Hamou-Lhadj, A., and Bousse, E. (2019). Model execution tracing: a systematic mapping study. *Software and Systems Modeling*, 18(6):3461–3485.

Jamshidi, M. (2008). *System of Systems Engineering: Innovations for the 21st Century*.

Levis, A. H. and Wagenhals, L. W. (2000). C4isr architectures: I. developing a process for c4isr architecture design. *Systems Engineering*, 3(4):225–247.

Maier, M. W. (1999). Architecting principles for systems-of-systems. *Systems Engineering*, 1(4):267 – 284.

Manoj Kumar, N., Goel, S., and Mallick, P. K. (2018). Smart cities in india: Features, policies, current status, and challenges. pages 1–4.

Manzano, W., Graciano Neto, V. V., and Nakagawa, E. Y. (2020). Dynamic-SoS: An Approach for the Simulation of Systems-of-Systems Dynamic Architectures. *The Computer Journal*, 63(5):709–731.

Neto, A., Spínola, R., and Travassos, G. (2010). Developing software technologies through experimentation: Experiences from the battlefield. pages 107–120.

Northrop, L., Feiler, P., Gabriel, R., Goodenough, J., Linger, R., Longstaff, T., Kazman, R., Klein, M., Schmidt, D., Sullivan, K., and Wallnau, K. (2006). *Ultra-Large-Scale Systems - The Software Challenge of the Future*.

Oquendo, F. (2016). Formally describing the software architecture of systems-of-systems with sosadl. In *2016 11th System of Systems Engineering Conference (SoSE)*, pages 1–6.

Wang, R. and Dagli, C. (2011). Executable system architecting using systems modeling language in conjunction with colored petri nets in a model-driven systems development process. *Systems Engineering*, 14(4):383–409.