How Much Does It Cost? A Simulation-Based Method for Cost Prediction in Systems-of-Systems Acquisition Processes

Valdemar V. Graciano Neto[1], Flávio E. A. Horita[2], Rodrigo P. dos Santos[3], Davi Viana[4], Mohamad Kassab[5]

[1]Universidade Federal de Goiás, Goiânia – GO – Brazil

[2]Universidade Federal do ABC, Santo André – SP – Brazil

[3]Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro – RJ – Brazil

[4]Universidade Federal do Maranhão, São Luiz – MA – Brazil

[5]Pennsylvania State University, Malvern – PA – United States

valdemarneto@inf.ufg.br, flavio.horita@ufabc.edu.br, rps@uniriotec.br

davi.viana@lsdi.ufma.br, muk36@psu.edu

Abstract. *Software economics, acquisition, and pricing are important concerns, in particular for Systems-of-Systems (SoS). SoS are alliances of independent software-intensive systems combined to offer holistic functionalities as a result of the constituents interoperability. SoS engineering involves separately acquiring constituents and combining them to form the SoS. Despite the existence of cost prediction techniques at Systems Engineering practice, predicting SoS acquisition costs at design-time should include: 1) an analysis of the minimum set of constituents that offer a 'good enough' result, and 2) an analysis of the compatibility between the constituents to deliver the expected result. The main contribution of this paper is proposing a novel simulation-based method for cost prediction in constituents acquisition process, while considering the effectiveness of constituents combination to offer the intended functionalities, and predicting the lowest configuration, at design-time. We adopt a simulation model to predict, at design-time, the results that shall be yielded by the constituents during SoS operation. Preliminary results point out the success of our method to predict such costs while still supporting a selection of the best architectural configurations.*

1. Introduction

Economic aspects are a pungent concern for both software production and software acquisition [Boehm and Sullivan 2000]. Software economics involves, in particular, an activity known as *software costing*, which consists of establishing a price for selling a software product (under the software vendor perspective) and assessing whether the price is fair considering the functionalities provided (from the client point of view). Software costing is actually required in software acquisition processes. Public governments open public announcements of software acquisition and software companies compete for selling their products. Companies establish prices for their software products and offer them. If they match the specification requirements with the lowest price, the government acquire their

software. However, with the emergence of smart cities, software pricing and acquisition processes have faced some additional challenges. Such process have involved, besides other concerns, the acquisition of multiple systems (e.g., Flood Monitoring Systems and Smart traffic systems) to form what is nowadays known as Systems-of-Systems (SoS[1]).

SoS comprise many independent software-intensive systems, known as constituents, that are combined to offer complex functionalities that could not be individually offered by their constituents. Since SoS depend on the compatibility among its constituents to achieve a cohesive mission, the design of a SoS should involve a careful selection of the participating constituents that exhibit the desired capabilities [Burton et al. 2014] and that best results to contribute to the accomplishment of the pre-established missions [Silva et al. 2015]. However, several candidate constituents may offer similar functionalities and thus it is important to consider other distinguishing factors such as the cost and predict how they will influence in the SoS holistic performance.

Acquisition of systems to be part of a larger set of interoperable systems is not a new trend, it has occurred since the 1970s in the USA, especially in the military domain [Acker 1983]. Satellites, airplanes, missiles, and systems have been purchased to interoperate for a long time during the last decades. However, the constituents are often individually acquired without (i) a thourough investigation on the value delivered when integrated within a larger system, (ii) a guarantee of functional compatibility, (iii) thorough investigation on the architectural configurations required to optimize the overall results, and (iv) a determination of the amount of constituents effectively needed to solve a problem. Evaluating costs and benefits at SoS context can be a complex task, since during its execution, a SoS can assume several distinct architectural configurations, which present different results that can influence the number of constituents required to be acquired, and the arrangement that should be maintained during SoS operation. Decisions made in the software development processes, especially in software architecture, have economic implications on the cost perspective. Therefore, it is important to investigate this economic aspect.

The main contribution of this paper is a novel simulation-based method to predict the optimal cost for SoS constituents acquisition. The method comprises three main steps: (1) the analysis of diverse architectural configurations a SoS can assume at runtime; (2) the selection of architectural configurations that offer the best combination of cost and performance; and (3) the assignment of an acquisition cost for that SoS based on a list of prices for each constituent system. Preliminary results reveal that our method is able to select the best architectural configurations, besides supporting a prediction of costs on the involved constituents. Such advance is important with the imminence smart cities (a type of SoS) and the respective government acquisition processes for SoS conception that may take place soon.

The paper is structured as follows: Section 2 presents the foundations to understand our proposal. Section 3 details our method, while Section 4 shows results of a preliminary evaluation. Section 5 discusses our results. Finally, Section 6 draws conclusions and indicates future work.

[1]For sake of simplicity, along this text, this acronym will be interchangeably used to express both singular and plural forms.

2. Background

SoS comprise a set of operational and managerial independent systems combined to offer larger functionalities that could not be individually delivered by any of them [Maier 1998]. Such complex functionalities are materialized as intended emergent behaviors, which can be intentionally engineered to accomplish a pre-defined set of missions [Rodriguez and Nakagawa 2017]. Individual missions are realized by constituent systems themselves whereas global missions of an SoS are accomplished through emergent behaviors [Silva et al. 2015]. SoS fulfill global missions by (i) performing assigned activities (individual missions) through constituents capabilities, and (ii) interactions among constituent systems leading to emergent behaviors.

SoS holds software architectures. A single software architecture comprises the fundamental structure of a software system, which comprises software elements, relations among them, and the rationale, properties, and principles governing their design and evolution [ISO 2011, Bass et al. 2012]. In turn, a SoS software architecture involves its fundamental structure, which includes its constituents and connections among them, their properties as well as those of the surrounding environment [Nielsen et al. 2015]. SoS software architectures are highly dynamic, i.e., they continuously change in response to addition, substitution, and deletion of constituents [Cavalcante et al. 2015]. In SoS software architectures, an *architectural configuration* is the current state and organization of an arrangement of interoperable software-intensive systems at a given point of time, also known as *coalition* in SoS domain. During its operation, a SoS software architecture can assume many different architectural configurations due to its *dynamic architecture* property. Each architectural configuration yields specific values about performance, reliability, and effectiveness. Such values can be collected through simulations, which enable an architect to anticipate, at design-time, the structure and behavior of a SoS before being deployed [Graciano Neto et al. 2018]. Once the best configurations are achieved, i.e., those systems that exhibit the best results with the lowest cost (the lowest number of constituents) are found, a self-healing mechanism can be triggered to maintain that coalition along the rest of the SoS operation, unless it occurs an emerging need of changing such structure. Therefore, coalitions can be predicted at design-time through simulations, and deployed to work later. Hence, the cost of system acquisition can be calculated in function of the predicted set of necessary (and enough) constituents, besides a margin of replacement (such as 10% of extra constituents) in case of defects or need of substitution.

Acquiring constituents to form such SoS depend on a twofold analysis: (i) the selection of constituents that offer the required set of capabilities necessary to fulfill the pre-established missions, and (ii) an assessment of the coalitions that offer the best results. Such results are often based on quality attributes, such as performance, and the available budget. Hence, constituents acquisition inherently involves a cost-benefit trade-off analysis, i.e., a balance between the advantages offered by a product and its associated cost.

Performance is one of the most important quality attributes to be analyzed in an acquisition process. At SoS context, performance concerns the results yielded by the synergy between the constituents systems to achieve the desired overall system goals[2].

[2] `http://www.sebokwiki.org/wiki/Architecting_Approaches_for_Systems_of_Systems`

Despite performance definition is not consensual for SoS context [Santos et al. 2014], for the scope of this paper, performance is defined as the degree of effectiveness of a coalition to accomplish a mission. This is measured as a relation between the effective number of data transported across a SoS architecture without losses and the total number of data provided as stimuli for feeding a simulation.

2.1. Related Work

SoS acquisition processes are often based on capability-based planning approaches, i.e., an optimization procedure that searches for a good solution that balances the set of desired capabilities and potential coalitions [Burton et al. 2014]. TLCM (Through Life Capability Management) [Urwin et al. 2010] and CapDEM [Robbins et al. 2005] are examples of approaches that rely on capability-based planning for predicting acquisition cost. However, those processes do not address an anticipation of the results exhibited by those coalitions measured in terms of quality attributes.

Burton et al. (2012) adopt a Model-Driven Engineering (MDE) approach, which includes domain-specific modeling languages to automatically generate potential solutions to the acquisition problem [Burton et al. 2012]. They progressed towards visualization techniques for the proposed solutions, and trade-off analysis for acquisition [Burton et al. 2014]. However, there is no focus on the results yielded by those potential solutions, specially considering quality attributes.

A recent work has invested on simulations for predicting attributes of a SoS software architecture at design-time [Graciano Neto et al. 2018]. In this approach, the authors specify a SoS software architecture using SoSADL models[3] [Oquendo 2016b], and automatically generating simulation models documented in DEVS [Zeigler et al. 2012]. After the assessment of multiple coalitions, the best configuration is elected. In the next section we show how such approach has been exploited for prediction of SoS acquisition cost.

3. A Simulation-Based Method to Support Constituents Acquisition for Systems-of-Systems Engineering

Figure 1 depicts our method that aims to help us predicting the selection of the best architectural configurations.

For determining the cost of system acquistion, the method starts with a list of consituent systems that goes through the following steps:

Step 1. SoS Architectural specification in SosADL. In this first step, an architecture of SoS is generated using SoSADL models;
Step 2. Model transformation execution. Having SosADL models produced, these are used as input for a model transformation to automatically generate simulation models specified in DEVS (a discrete-event simulation formalism)[4];
Step 3. Simulation execution. DEVS models produced in Step 2 are executed using MS4ME platform. Such Eclipse-based environment enables (i) the visualization of messages exchanged between constituents during SoS execution, (ii) dynamic architecture,

[3]SosADL is an architectural description language specially created for SoS domain.

[4]Details about the model transformation and how SosADL and DEVS models are mapped between them are not the focus of this paper and can be found in [Graciano Neto et al. 2018].

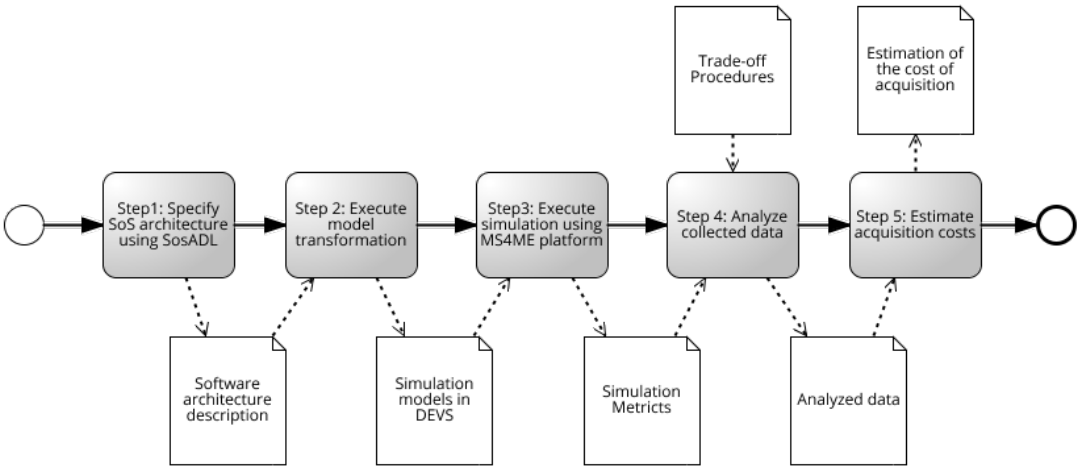

Figure 1. A Simulation-Based Method to Support Constituents Acquisition for Systems-of-Systems Engineering

and (iii) the measurement of pre-established metrics related to quality attributes;
Step 4. Coalitions analysis. After metrics collection, it is possible to analyze values delivered by coalitions through a trade-off procedure, supporting the decision of the coalition that offers the best combination between cost and benefits;
Step 5. Cost estimation. Once the constituents involved in that selected coalition are already defined, a table of prices can be used to estimate (with precision) the cost of acquisition for that set of constituents.

4. Evaluation

We conducted a pilot study to evaluate our method in supporting a precise prediction of acquisition cost in function of the selection of coalitions through simulations at design-time. A Flood Monitoring SoS (FMSoS) was adopted as an application scenario in this work. This SoS monitors rivers crossing urban areas, which pose great danger of floods in rainy seasons, potentially damaging property, lives, and possibly spreading diseases. FM-SoS notifies possible emergency situations to residents, businesses owners, pedestrians, and drivers located near the flooding area, and also to governmental entities and emergency systems. Its mission comprises ***the emission of flood alerts.*** FMSoS is composed of three different types of constituents:

1. **Smart sensors**, which are fixed embedded systems monitoring flood occurrences in urban areas, located on river edges;
2. **Gateways**, which gather data from constituents and share them with other systems;
3. **Crowdsourcing platforms**, which are mobile applications used by citizens for real-time communication of water level rising; danger level is a pre-defined value (between 1 and 6, 1 being no risk, and 6 being flood effectively occurring) that can be classified by a human user according to what he/she observes.

We provide a detailed overview on the execution of each step of our method for the application case (FMSoS).

Step 1. SoS Architectural specification in SosADL. In this step, SoSADL models were elaborated to register the FMSoS architecture. We modeled one of each type of constituent in SosADL, and replicated them.

Step 2. Model transformation execution. We run the model transformation and produced the simulation models. These were later deployed in MS4ME environment.

Step 3. Simulation execution. After deploying the DEVS models, we executed the simulation. It started with a configuration of constituent systems (i.e., four smart sensors, one gateway, and no crowdsourcing system). Along the execution, we exploited the dynamic architecture ability to include variations of the systems. This type of investigation enabled us to study the SoS behavior evolution based on its structural changes.

Step 4. Coalitions analysis. For this context, we compared coalitions considering the number of constituents of a coalition and their effectiveness to transport the data used to feed them. This analysis was done because the triggering of an emergent behavior is only possible as a result of the data exchange between constituents. Hence, the more successful a SoS architecture is to correctly transport data, the more effective it is to complete a given mission through its correspondent emergent behavior. Figure 2 plots the results of our analysis, taking into account (i) the percentage of the data fed to sensors that were correctly transmitted along the SoS architecture until the gateways considering the variation in the number of constituents, and (ii) the percentage of flood alerts that were triggered. It was observed that data loss increased with the number of sensors, reducing both the reliability of data transmission and triggered alerts (Point 2). This loss was alleviated by increasing the number of gateways, which increased the numbers of transmission rate and triggered alerts. When the architecture configuration had 40 constituents (Point 3), i.e., 30 sensors and 10 gateways (without considering mediators), the number of crowdsourcing platforms was increased as well. However, despite the expectation, increasing the number of crowdsourcing platforms neither increase the transmission rate, nor the number of alerts triggered because of the bottleneck of the gateways. Results improved again when the number of crowdsourcing platforms was fixed at 20 (Point 4), and the number of gateways was increased to 20 (Point 5), with 30 sensors, 20 gateways, and 20 crowdsourcing platforms (70 constituents, without considering mediators). It was also possible to observe that the rate of alerts correctly triggered was close to the rate of data effectively transmitted[5].

Good results occurred when FMSoS has many constituents, but these results are not better than when FMSoS has only five constituents. Hence, unless there is a situation in which a geographic area to be covered is too large, using a small number of constituents can achieve similar results as using a large number, at least for this domain, these configurations defined, and these types of constituents.

Step 5. Cost estimation. After selecting the coalition depicted in Point 1 of Figure 2, the cost estimation procedure was performed. We estimated prices for each type of constituent used in FMSoS in monetary units. As crowdsourcing systems are owned by population, only smart sensors and gateways were considered[6]. XBee technology is

[5]Results replicated from a previous study [Graciano Neto et al. 2018].

[6]SoS also depend on a type of element called mediator, which enables the forwarding of data between constituents, such as ZigBee wireless links. One mediator is required between each pair of constituents. We did not consider them in this study.

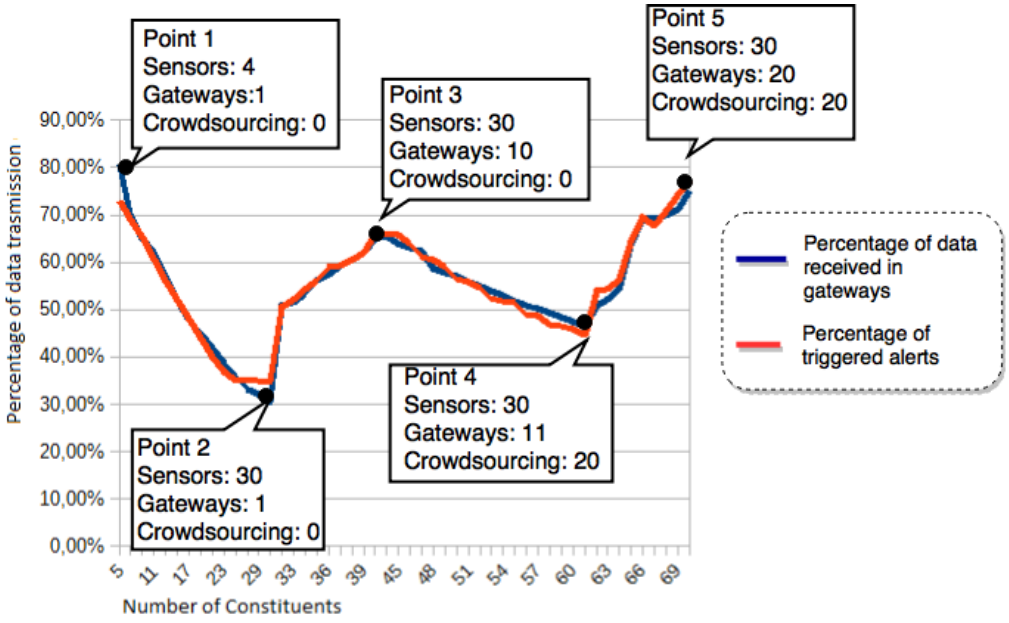

Figure 2. Relation between percentage of data received in gateways and alerts triggered [Graciano Neto et al. 2018].

one instance of smart sensor brand, whilst gateways can be materialized by an industrial computer linked to the Internet [Oquendo 2016a]. A smart flood sensor can be acquired from prices that varies from U$ 57.77 to U$ 79.99[7]. In turn, prices for gateways vary from U$ 28.95 to U$ 61.33[8].

Table 1 shows a comparison between realistic estimation of cost acquisition for smart sensors and gateways of four different brands. For this study, we assume that all of them deliver the same performance.

Table 1. Comparison of prices between different constituent brands.

	Brand A	*Brand B*	*Brand C*	*Brand D*
Smart Sensor	$57.77	$64.99	$66.99	$79.99
	Brand E	*Brand F*	*Brand G*	*Brand H*
Gateway	$28.95	$39.95	$49.95	$61.33
Final Best Price for the selected coalition using cheapest prices among brands.				$260.03

The lowest price found, as presented in Table 1, was $260,03. However, it is important to highlight that the acquisition costs could reach $2312,1 (Point 5 in Figure 2: 30 sensors and 20 gateways) if we have not performed an analysis that revealed a similar result between a small coalition and a large coalition. While this value is relatively high, our method demonstrates that the acquisition costs can be reduced in almost 90% ($2312,1 in relation to $260,03) in a SoS acquisition process due to a careful analysis of the results yielded by different coalitions. Moreover, in an acquisition process for military domain,

[7]https://goo.gl/9Lu5Ug
[8]https://goo.gl/cNxuFh

whose the order of magnitude of prices reaches millions of dollars, this type of procedure can be even more valuable.

5. Discussion

Our simulation-based method covers characteristics that are essential for SoS domain and that are not covered by other lines of studies on SOS, such as the dynamic architecture of the SoS, and analysis of emergent behaviors [Oquendo 2016b, Oquendo 2017]. Moreover, other proposals have been more focused on optimization problems to find, within the expected spectrum of constituent capabilities, the minimum set of constituents, without a thorough analysis of performance [Burton et al. 2014]. Our method analyzes the results delivered by the different coalitions according to a set of metrics (pre-defined in the context of the SoS architectural analysis approach), allowing a trade-off on metrics related to the quality and cost attributes of SoS software architecture.

This work also contribute to previous works on the role of architects of software ecosystems [Weinreich and Groher 2016, Amorim et al. 2017]. Within software ecosystems, the architect is resposible to define the optimal strategy of product because he knows the customers' needs and priorities. Results obtained in the evaluation of our method provide valuable metrics, as well as an organization of optimal arrangement of systems that would support a decision-making for software architects. In other words, they can make decisions considering not only customers' needs but also the lowest cost and best performance of systems.

We highlight the following threats to the validity of our conclusions: transformation correctness, human failure during estimation of prices, and choice of the best coalition. The same model transformation has already been used to make dozens of transformations between SosADL and DEVS models for two different domains: smart cities and space. Therefore, this threat is relieved by the number of studies that have already used such transformation. In addition, although formal proofs of its correctness have not been conducted, it generates correctly specified simulations every time. Such result is reliable because in the DEVS formalism a single erroneous instruction may make the simulation execution unfeasible, causing it to crash or even preventing its execution. From the point of view of human failure, there are some points in the process that are subject to failures, such as the observance and collection of metrics, as well as the choice among prices. A study with real data was performed on a small scale. However, results indicate the feasibility of reproducing it on a larger scale. Moreover, automation processes can be conducted to avoid human errors and enable the study of larger instances.

6. Final Remarks

Cost is a primary driver to decide whether to build a SoS or to create a new specialized system [Johnson 2015]. Moreover, cost is a relevant economic aspect of systems. Our simulation-based method enables to evaluate the performance of different arrangements of constituents and decide which constituents should be bought to form that SoS, offering the best performance and lowest cost.

After conducting a pilot study, we concluded that using a small number of constituents could achieve the same results as using a large number of constituents. Owing to such information, it is possible to anticipate which constituents are effectively necessary

to build a SoS, and predict the budget necessary to acquire them. The acquisition and construction of a SoS involves acquiring hardware in which software will be deployed with specific capabilities to realize the intended emergent behavior. In this paper, we exploited (i) the prediction of software architectures of a SoS at design-time, (ii) the prediction of different coalitions such architecture could assume at runtime, and (iii) the results that each one of such coalitions yield to suppport (iv) a prediction of cost of acquisition of the corresponding hardware necessary to support the existence of that SoS.

With the emergence of smart cities, software-intensive SoS will become highly open and dynamic. As such, dealing with acquisition cost comprises the prediction of their results at design-time, and the acquisition of the hardware in which the correspoding software will be deployed. Next steps of investigation include the prediction of software acquisition and software development under man-hour metric and function points prediction for development of software for constituents. Besides, our approach does not currently explore the diversity of constituents, i.e., we did not conduct a study in which products of different brands are benchmarked in order to choose not only the best coalitions, but also brands that offer lowest prices and better performance. This shall also be covered in forthcoming advances on this research. Other future work lines include (i) comparison among coalitions through the substitution of constituents that offer the same capability for better decision-making between different brands, (ii) adoption of co-simulation to accurately reproduce the scenarios required for other quality attributes such as security [Hachem et al. 2016], and (iii) establishment of a mechanism for automation of the cost estimation through the integration between the simulator, a mechanism for querying and comparing market prices, and some model-checker mechanism to automatically deliver the best coalition, without the need to manually collect and analyze data. We also consider that, for large volumes of data, we can apply search-based software engineering to support the selection of constituents from criteria related to technical and economic aspects of software. Nevertheless, we highlight the importance of the results achieved until now and the seminal nature of our solution for SoS domain.

References

Acker, D. D. (1983). Defense systems acquisition review process: A history and evaluation. Technical report, Defense Systems Management Coll Fort Belvoir Va.

Amorim, S. S., McGregor, J. D., de Almeida, E. S., and von Flach G. Chavez, C. (2017). The architect's role in software ecosystems health. In *WASHES*, pages 1–4.

Bass, L., Clements, P., and Kazman, R. (2012). *Software Architecture in Practice*. Addison-Wesley Professional, Indianapolis, Indiana, USA, 3rd edition.

Boehm, B. W. and Sullivan, K. J. (2000). Software economics: A roadmap. ICSE '00, pages 319–343, New York, NY, USA. ACM.

Burton, F. R., Paige, R. F., Poulding, S., and Smith, S. (2014). System of systems acquisition trade-offs. *Procedia Computer Science*, 28:11–18.

Burton et al. (2012). Solving acquisition problems using model-driven engineering. In *ECMFA*, volume 7349, pages 428–443, Lyngby, Denmark. Springer.

Cavalcante, E., Batista, T. V., and Oquendo, F. (2015). Supporting dynamic software architectures: From architectural description to implementation. In *WICSA 2015*, pages 31–40, Montreal, Canada. IEEE.

Graciano Neto, V. V., Garcés, L., Guessi, M., Paes, C., Manzano, W., Oquendo, F., and Nakagawa, E. Y. (2018). ASAS: An approach to support simulation of smart systems. In *51th HICSS*, pages 5777–5786, Big Island, Hawaii, USA. IEEE.

Hachem, J. E., Pang, Z. Y., Chiprianov, V., Babar, A., and Aniorté, P. (2016). Model driven software security architecture of systems-of-systems. In *23rd Asia-Pacific Software Engineering Conference*, pages 89–96, Hamilton, New Zealand. IEEE.

ISO (2011). ISO/IEC/IEEE 42010:2011 - Systems and software engineering – Architecture description. *ISO*, pages 1–46.

Johnson, S. B. (2015). System health management. In Rainey, L. B. and Tolk, A., editors, *Modeling and Simulation Support for System of Systems Engineering Applications*, pages 131–144. Wiley, Hoboken, New Jersey, USA.

Maier, M. W. (1998). Architecting principles for systems-of-systems. *Systems Engineering*, 1(4):267–284.

Nielsen, C. B., Larsen, P. G., Fitzgerald, J., Woodcock, J., and Peleska, J. (2015). Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions. *ACM Computing Surveys*, 48(2):18:1–18:41.

Oquendo, F. (2016a). Case study on formally describing the architecture of a software-intensive system-of-systems with sosadl. In *SMC 2016*, pages 2260–2266, Budapest, Hungary. IEEE.

Oquendo, F. (2016b). Formally Describing the Software Architecture of Systems-of-Systems with SosADL. In *11th Annual System of Systems Engineering (SOSE 2016)*, pages 1–6, Kongsberg, Norway. IEEE.

Oquendo, F. (2017). Architecturally describing the emergent behavior of software-intensive system-of-systems with SosADL. In *12th SoSE*, pages 1–6, Waikoloa, USA. IEEE.

Robbins, W., Lam, S., and Lalancette, C. (2005). Towards a collaborative engineering environment to support capability engineering. In *Proceedings of the 2005 INCOSE International Symposium*, pages 211–221, Rochester, NY, USA.

Rodriguez, L. M. G. and Nakagawa, E. Y. (2017). A process to establish, model and validate missions of systems-of-systems in reference architectures. In *SAC 2017*, pages 1765–1772, Marrakech, Morocco. ACM.

Santos, D. S., Oliveira, B., , Guessi, M., Oquendo, F., Delamaro, M., and Nakagawa, E. Y. (2014). Towards the evaluation of system-of-systems software architectures. In *8th WDES*, pages 53 – 57, Maceió, Brazil. SBC.

Silva, E., Batista, T., and Cavalcante, E. (2015). A mission-oriented tool for system-of-systems modeling. In *3th SESoS*, pages 31–36, Florence, Italy. IEEE.

Urwin, E. N., Pilfold, S. A., and Henshaw, M. (2010). Through life capability management: benefits and behaviours. In *International Conference on Contemporary Ergonomics and Human Factors*, pages 153–162. CRC Press.

Weinreich, R. and Groher, I. (2016). The architect's role in practice: From decision maker to knowledge manager? *IEEE Software*, 33(6):63–69.

Zeigler, B. P., Sarjoughian, H. S., Duboz, R., and Souli, J.-C. (2012). *Guide to Modeling and Simulation of Systems of Systems*. Springer-Verlag, London, United Kingdom.