Towards a Conceptual Model for Software-Intensive System-of-Systems

Marcelo Benites Gonçalves¹,², Everton Cavalcante²,³, Thais Batista², Flavio Oquendo³, Elisa Yumi Nakagawa³

¹Department of Computer Systems, University of São Paulo, São Carlos, Brazil
²Department of Informatics and Applied Mathematics, Federal University of Rio Grande do Norte, Natal, Brazil
³Institut de Recherche en Informatique et Systèmes Aléatoires, Université de Bretagne-Sud, Vannes, France

Abstract—Software systems have become increasingly large and complex, often resulting from the integration of several operationally independent systems, thus leading to a new class of systems called Systems-of-Systems (SoS). Moreover, SoS have currently become essentially software-dependent systems, thus arising the Software-intensive Systems-of-Systems (SiSoS). However, in spite of their relevance, there is a lack of a common understanding of what exactly SiSoS are, their characteristics and types, as well as other elements that surround them. In this context, the main contribution of this work is to provide a conceptual model that provides a comprehensive understanding about them. For this purpose, we have extended the ISO/IEC/IEEE 42010 International Standard by considering related studies and previous taxonomies about SoS. In order to illustrate the use of our model, two software-intensive systems were analyzed and discussed. By establishing a common understanding about SiSoS, results achieved until now have showed us that this model can provide an important support to the research on SiSoS and, as a consequence, it can contribute to the development of these systems.

Index Terms—Software-intensive system, system-of-systems, SiSoS, conceptual model.

I. INTRODUCTION

In the last years there has been a growing interest in the research and development of complex systems resulted from the integration of other independent and heterogeneous systems, called Systems-of-Systems (SoS). The SoS concept started to gain its popularity mainly in military systems of the American Department of Defense (DoD) [1]. SoS was initially applied as a strategy of how to reach goals or to deliver unique capabilities that are the result of a collaborative work of a dynamic set of existing systems [1] [2] [3]. Furthermore, complex, large scale systems for several society’s needs, such as healthcare, logistics, energy, and transportation, started to meet the SoS concept [4].

According to the ISO/IEC/IEEE 42010 International Standard, a software-intensive system is any system in which software essentially influences the design, construction, deployment, and evolution of the system as a whole to encompass individual applications, subsystems, systems-of-systems, product lines, product families, whole enterprises and other aggregations of interest [5]. By following the natural trend of complex, large-scale systems to be more software dependent, SoS tend to be also software-dependent, thus becoming the so-called Software-intensive SoS (SiSoS) [6].

In this scenario, we have observed that there is no consensual and structured description for SiSoS that allows to clarify the boundaries and the sort of problems related to this class of system. Despite the existence of relevant descriptions and taxonomies about SoS, they do not encompass SiSoS and provide a view of SoS from the perspective of system engineering disciplines. Furthermore, the lack of a common understanding about SiSoS hampers the adoption of better strategies to develop and maintain them. It is also difficult to classify or even to differentiate SiSoS from other categories of systems that have similar aspects with SoS, such as large-scale, complex, and distributed systems. Without a consensus about this class of systems, SiSoS are frequently developed without the “SoS” label [7].

In this context, this paper presents a conceptual model for SiSoS aiming at enabling a consensual understanding about this class of systems and related concepts. In order to establish it, related studies and previous taxonomies about SoS were investigated and all characteristics and definitions were crossed and analyzed within a software-intensive systems perspective as an extension of the ISO/IEC/IEEE 42010 Standard. The proposed model is not just a review of the existing literature about SoS, but it represents an effort towards the standardization of the concepts related to SiSoS. Moreover, such a model can be useful to support the research on SiSoS by being both a source of consensual knowledge about this class of systems and an useful resource for analyzing software-intensive systems under the SiSoS perspective. Aiming at illustrating the use of our proposal, we have analyzed two software-intensive systems by using the conceptual model as a reference. Through the performed analyses, it was possible to verify the applicability of the model for adequately identifying if a software-intensive system is a SiSoS or not and to which category of SiSoS it can belong.

The remainder of this paper is organized as follows. In Section II we present the background of our research. In Section III we present our conceptual model for SiSoS and its key concepts. Section IV illustrates the use of the proposed model. Finally, in Section V we summarize our contributions and discuss perspectives for future work.
II. BACKGROUND

The initial efforts to establish definitions and taxonomies for SoS were initiated in the context of the systems engineering. The first taxonomy for SoS was proposed by Maier [2] in 1990’s in which three SoS basic types (virtual, collaborative, and directed) and five main characteristics (operational independence, managerial independence, evolutionary development, emergent behavior and geographic distribution) are specified. With the evolution of the SoS field, several definitions for SoS in different contexts were proposed later. In this perspective, some studies [3] [8] [9] [10] have analyzed these definitions available in the related literature in order to understand their differences and commonalities. These studies ratified the characteristics initially proposed by Maier as the consensual ones for SoS.

Other SoS taxonomies are typically focused on the support to SoS development [11] [12]. For instance, DeLaurentis [11] proposed a taxonomy for designing SoS with decision-making support. This taxonomy proposes three axes (i.e., key dimensions) for SoS classification, namely system type, connectivity, and control. Despite the support for decision-making on the SoS design, it is not focused on software-intensive systems. For example, its system type dimension captures the technological level of the SoS in which hardware-oriented and human-based systems are also accepted.

In the same perspective, there are ontologies developed in the SoS context that have other purposes than the SoS understanding and characterization, such as systems design methodologies [13], knowledge management [14], and interoperability management [15]. In this context, Sarder and Ferreira [16] propose a preliminary functional domain ontology on systems engineering context and they highlight the need of a general ontology for SoS in this context.

Despite the existence of a consensus about the typical features of SoS, there is no common standard for classifying this class of systems or a standard for SiSoS. In this context, SiSoS have emerged without a consensual understanding as they do not still have a set of basic features for their characterization. The studies about SoS are more focused on systems engineering and they are not completely adequate for tackling SiSoS. Due to this gap, the formalization of a model that encompasses the knowledge related to SiSoS is quite important to enable a consensus about this class of system.

III. SOFTWARE-INTENSIVE SOs CONCEPTUAL MODEL

In this section we present a conceptual model for SiSoS that does not only support their characterization, but it also allows to differ SiSoS from other categories of systems. This distinction is quite important due to the existence of several systems that have similar characteristics with SiSoS, but cannot be classified as such [17]. For example, due to the typical geographical distribution of the constituent systems, a SiSoS can be viewed as a subclass of distributed systems, but not all distributed systems will be a SiSoS.

Considering that SiSoS represent a subclass of SoS, we have built our model upon the related literature in order to make it coherent with the existing knowledge about SoS. At first, we have considered studies that address consensual characteristics and definitions related to SoS [2] [9] [18]. These concepts were analyzed within the software-intensive systems perspective, so that our strategy was to extend the ISO/IEC/IEEE 42010 International Standard [5], which provides a conceptual framework for the creation, analysis, and specification of architectures of software-intensive systems. We have chosen it as a basis for our conceptual model since it is a standard and it also addresses important elements regarding software-intensive systems, such as the notions of system and its stakeholders, purpose, and environment. Moreover, we have not considered existing taxonomies and ontologies that described SoS because none of them properly address SiSoS [11] [12].

Fig. 1 depicts the elements of our model and the relationships among them, whereas Table I presents the definition of the key concepts required to analyze a system by using the model. At first, as a SiSoS is an SoS, a SiSoS also exists to accomplish a global mission, which represents its major goals. A SiSoS is composed of constituent systems, which contribute to the accomplishment of the global mission of the SiSoS and they can have managerial independence and an individual mission. Moreover, constituent systems have operational independence and present the ability of connecting to each other (connectivity [17] [20]), thus enabling their cooperation in order to yield emergent functionalities, the so-called emergent behavior. In addition, both a SiSoS and its constituent systems can have their respective stakeholders (SoS and constituent systems stakeholders), which have interests in their missions.

Considering the software-intensive systems perspective, a SiSoS is an SoS in which software is the dominant factor (i.e., it exhibits a software dominance) in its composition. Despite a SiSoS can integrate human-based, hardware-based or software-based constituent systems [11], the software constituent systems must be the dominant factor for the SiSoS operation. In other words, the types of constituent systems influence whether an SoS will be software-intensive or not. Therefore, for SiSoS, the set of integrated software constituent systems justifies the software dominance, so that a SiSoS cannot operate without them.

SiSoS are typically complex, large-scale systems whose functionalities and purposes can change and dynamically evolve, i.e., they present an evolutionary development [2] [3]. As a consequence, the architecture exhibited by a SiSoS is dynamic and reactive since it must evolve according to the dynamic collaboration of constituent systems that are also changing. Moreover, the representation and simulation of these architectures allow to understand possible emergent behaviors, even the not desired ones.

Some of the concepts are determinant to categorize a SiSoS. One of them is the central authority of the SiSoS that determines how the constituent systems cooperate, thus resulting in four basic categories of SiSoS, namely virtual, collaborative, acknowledged, and directed, as summarized in Table II. Another one is the awareness level, which stands
for the degree in which the constituent systems are aware of their participation in the SiSoS as whole. In virtual SiSoS, the constituent systems are completely unaware of their participation in the system. For the other categories of SiSoS (collaborative, acknowledged, and directed), the constituent systems are aware of such a participation.

By considering the use of the model for analyzing systems in the SiSoS context, we consider that a system can be characterized as a SiSoS if all of the key characteristics presented in Table I are encompassed. For example, the operational independence is adequately encompassed if the constituent systems have at least the ability of operating in an independent way, even when they do not exercise such an ability. After identifying that the system is a SiSoS, its category is determined according to the definitions presented in Table II. Therefore, the concepts presented in the proposed model enable to identify if a software-intensive system is a SiSoS or not and to which category of SiSoS it belongs.

IV. APPLICATION

In order to illustrate the use of our proposal, we have analyzed two software-intensive systems by using the key concepts of the model. This analysis aimed to clarify how systems can be understood under the SiSoS perspective and also to assess how our model can be useful for classifying SiSoS and differentiating this class of systems from other ones.

Table III summarizes the differences between the analyzed systems according to the key characteristics of a SiSoS. The
The constituent systems are independently managed in a distributed and uncoordinated environment where the mechanisms to maintain the whole SoS are not evident. Therefore, by this set of aforementioned interoperability were developed and employed as GEOSS-registered standards. Therefore, by this set of aforementioned means, the authority is based upon negotiation between the constituent systems and the SoS as a whole.

A. Global Earth Observation System of Systems (GEOSS)

The Global Earth Observation System of Systems (GEOSS) is a global project that encompasses over 60 nations and is constituted of integrated systems that create information for environmental decision-making. GEOSS monitors several environments and its related conditions (e.g., climates, crops, forests, deserts) and their changes/effects, as well as potential impacts [21]. GEOSS has as a basic principle the adoption of international standards and common interoperability arrangements with a service-oriented architecture (SOA) approach [22]. Together, the GEOSS architectural principles and a cross-cutting discovery tool, called GEOSS clearinghouse, enable several organizations to cooperate in realizing the GEOSS [23].

At first, GEOSS is a software-intensive system (i.e., it has software dominance) because it is mainly composed of software-based Earth observation systems, such as satellites and sensors. Its global mission is to be a global public system that allows the analysis and exchange of Earth observation data for a wide range of users for the benefit of society. The GEOSS constituent systems have operational independence since they come from different development sources, each one being subordinated of its own source (country, organization, group, etc.). Moreover, GEOSS includes shared data provided by different, geographically distributed sources. Regarding its development, GEOSS has an evolutionary development based on a SOA perspective in which the constituent systems and the structure of collaboration can evolve according to the GEOSS needs.

In the complex Earth environment, crossed information that emerges from GEOSS cannot be delivered by the constituent systems in an isolated manner, thus characterizing the emergent behavior. The connectivity is a fundamental concern in this scenario, so that a set of standards and methods for interoperability were developed and employed as GEOSS-registered standards. Therefore, by this set of aforementioned characteristics, GEOSS can be regarded as a SiSoS. Furthermore, GEOSS can be classified as a collaborative SiSoS because it has a central effort/authority that only enables the integration of the constituent systems by offering development support and promoting common technical standards, so that the constituent systems voluntarily collaborate with the GEOSS and its purposes.

B. WSN-Based Flood Monitoring System

Wireless sensor networks (WSNs) are composed of motes, which are tiny hardware/software platforms equipped with an embedded CPU, low power wireless networking capabilities, and simple sensors [24]. WSNs have been increasing employed in several real-world applications, such as flood monitoring in urban areas [25]. During rainy seasons, floods are challenging to urban centers traversed by large rivers due to material, human, and economic losses in flooded areas. In order to minimize these problems, a flood monitoring system can support the monitoring of urban rivers and create alert messages to notify authorities and citizens about the risks in case of an imminent flood, for example. This system is mainly composed sensors spread in the proximities of a river to monitor physical and/or environmental conditions (e.g., water level and flow rate of the river). Data gathered by these sensors are forwarded through wireless network connections to a base station gateway that analyzes such data and then triggers alerts if a flood condition is detected.

This flood monitoring system can be viewed as a software-intensive system mainly due to its software dominance, which is related to the role played by supporting software platforms. Moreover, due to their geographical distribution, its constituent parts are connected and collaborate towards the accomplishment of the global mission of the system, i.e., to monitor the river, to detect a risk of flood, and to trigger warning messages. These functionalities provided by the system as a whole arise from the collaboration among the constituent elements, which are not able to provide them if they are separately considered, thus encompassing the so-called emergent behavior of the system.

Despite this system presents the aforementioned features in common with a SiSoS, it cannot be classified as such. First, not all components of the system have operational independence.

<table>
<thead>
<tr>
<th>Category</th>
<th>Definition</th>
<th>Level of central authority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual</td>
<td>The constituent systems are independently managed in a distributed and uncoordinated environment where the mechanisms to maintain the whole SoS are not evident.</td>
<td>Nonexistent.</td>
</tr>
<tr>
<td>Collaborative</td>
<td>The constituent systems voluntarily collaborate more or less in order to address shared or common interests.</td>
<td>The authority offers standards to enable the collaboration of the constituent systems.</td>
</tr>
<tr>
<td>Acknowledged</td>
<td>The goals, management, resources, and central authority of the SoS are all recognized, but the constituent systems still retain their independent management.</td>
<td>The authority is based upon negotiation between the constituent systems and the SoS as a whole.</td>
</tr>
<tr>
<td>Directed</td>
<td>The constituent systems can have their operational and managerial independence, but their behavior is subordinated to the central authority and its purposes.</td>
<td>The authority and its specific main purposes are evident and drive the constituent systems.</td>
</tr>
</tbody>
</table>
Although the individual motes can continue providing sensed data out of the system as a whole, the base station gateway is not able to provide their functionalities when disassembled from the system, as it depends upon data provided by sensors for performing analyses about floods. Furthermore, there is a lack of a full support for an evolutionary development, an inherent characteristic for any SiSoS. For instance, the sensor nodes of the system are assigned to rigid roles, so that changing the function of a sensor node often requires on-site maintenance, which is a difficult and time-consuming activity.

V. CONCLUSIONS

Computational systems have evolved at some point in which the notion of holism must be over the traditionally applied reductionism, i.e., the resulted systems are more than the sum of its constituent parts. Given this scenario, in the last years there has been a growing interest in the research and development of SiSoS, which are complex software-intensive systems resulted from the integration of other independent, heterogeneous systems. Despite this interest, SiSoS have been emerged without a consensual understanding since they do not still have a set of basic features for their characterization. In this context, the main contribution of this work was to present a comprehensive conceptual model that supports the analysis and classification of this class of systems. In order to built this model, we have considered an extension of a standard for software-intensive systems aligned with the typical features of SoS found in the existing literature.

It is noteworthy that the proposed model is not just a review of the existing literature about SoS, but it represents an effort towards the standardization of the concepts related to this class of systems and, by extension, to SiSoS. Therefore, this model can useful to support the research on SiSoS by being a source of consensual knowledge about this class of systems and an useful resource for analyzing software-intensive systems under the SiSoS perspective. We also hope to contribute to the development and research on these systems by providing means to adequately understand and classify them. In this perspective, with the adoption of a common conceptual model for SiSoS, some benefits are expected, such as: (i) promoting the discussion and the consensus about SiSoS on software engineering disciplines; (ii) supporting the study and development of new software engineering proposals for SiSoS; (iii) formalizing and improving the understanding about SiSoS by abstracting their main features, and; (iv) clarifying SiSoS requirements.

TABLE III
BRIEF COMPARISON OF KEY CONCEPTS BETWEEN THE ANALYZED SOFTWARE-INTENSIVE SYSTEMS

<table>
<thead>
<tr>
<th>Key concepts</th>
<th>GEOSS</th>
<th>WSN-Based Flood Monitoring System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global mission</td>
<td>To be a global public infrastructure that generates data related to Earth observation</td>
<td>To monitor a river to detect risk of flood and to trigger warning messages</td>
</tr>
<tr>
<td>Software dominance</td>
<td>Majority of software embedded systems (e.g., satellites, sensors, etc.)</td>
<td>Software embedded systems</td>
</tr>
<tr>
<td>Operational independence</td>
<td>The constituent systems independently operate in its own environment over the Earth</td>
<td>Base station gateway is not operationally independent, thus not being able to provide their functionalities out of the system</td>
</tr>
<tr>
<td>Evolutionary development</td>
<td>The systems evolve with the collaboration of different organizations, development groups, and partner nations that act as solution providers</td>
<td>Lack of full support for evolution upon changes on requirements and/or at runtime</td>
</tr>
<tr>
<td>Emergent behavior</td>
<td>The complex crossed information that emerges from the GEOSS cannot be delivered by the constituent systems in an isolated manner</td>
<td>Functionalities provided by the system cannot be individually offered by the constituent parts</td>
</tr>
<tr>
<td>Geographical distribution</td>
<td>The constituent systems are geographically distributed over the Earth, thus exchanging only information</td>
<td>Motes and gateway are geographically distributed</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Promoted through a set of standards and methods for interoperability provided by the GEOSS community</td>
<td>Motes and gateway are connected through wireless network connections</td>
</tr>
</tbody>
</table>

SiSoS Category | Collaborative | --- |
REFERENCES


