Establishment of an ontology for Systems-of-Systems

Gabriel Abdalla
Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)
Establishment of an ontology for Systems-of-Systems

Master dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. FINAL VERSION
Concentration Area: Computer Science and Computational Mathematics
Advisor: Prof. Dr. Elisa Yumi Nakagawa

USP - São Carlos
October 2017
Abdalla, Gabriel

Establishment of an ontology for Systems-of-Systems / Gabriel Abdalla; orientadora Elisa Yumi Nakagawa. -- São Carlos, 2017.
76 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2017.

1. Systems-of-Systems. 2. Ontology. 3. OntoSoS. 4. Terminology. I. Nakagawa, Elisa Yumi, orient. II. Título.
Gabriel Abdalla

Estabelecimento de uma ontologia para Sistemas-de-Sistemas

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. VERSÃO REVISADA

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Elisa Yumi Nakagawa

USP - São Carlos
Outubro de 2017
ABSTRACT

ABDALLA, G. Establishment of an ontology for Systems-of-Systems. 2017. 76 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

Systems-of-Systems (SoS) represent an emerging research field in the Software Engineering area. In particular, SoS refer to systems that make possible the interoperability of distributed, complex systems, cooperating among them to reach a common mission. Several SoS have already been developed and used, but there is no consensus about diverse terms and concepts in this field, what can make difficult the communication among different stakeholders involved in the development and evolution of SoS, besides lacking of a standardization and common understanding among researchers and practitioners. This Master’s project established OntoSoS, an ontology to formalize terms and concepts in the SoS field, expliciting and allowing sharing and reuse of knowledge contained in such ontology. As a result, this project intends to contribute to the field of SoS, also supporting activities related to SoS Engineering. It is also expected that this ontology can serve as a learning material in courses related to SoS.

Keywords: Systems-of-Systems, Ontology, OntoSoS, Terminology.
RESUMO

ABDALLA, G. Estabelecimento de uma ontologia para Sistemas-de-Sistemas. 2017. 76p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

Sistemas-de-Sistemas (do inglês, Systems-of-Systems ou simplesmente SoS) representam um campo emergente de pesquisa na Engenharia de Software. Em particular, SoS referem-se a sistemas que possibilitam a interoperabilidade de sistemas complexos, distribuídos, cooperando entre si para atingir uma missão comum. Diversos SoS têm sido desenvolvidos e utilizados, mas não há um consenso sobre os diversos termos e conceitos nesse campo, o que pode dificultar a comunicação entre os diferentes interessados envolvidos no desenvolvimento e evolução dos SoS, além da falta de padronização e entendimento comum entre pesquisadores e profissionais. Este projeto de Mestrado estabeleceu a OntoSoS, uma ontologia para formalizar termos e conceitos no campo de SoS, explicitando e permitindo o compartilhamento e reúso do conhecimento contido na ontologia. Como resultado, este projeto pretende contribuir para o campo de SoS, auxiliando também nas atividades relacionadas à Engenharia de SoS. É também esperado que essa ontologia possa servir como um material de ensino em cursos relacionados à Engenharia de SoS.

Palavras-chave: Sistemas-de-Sistemas, Ontologia, OntoSoS, Terminologia.
## Contents

1 Introduction

1.1 Contextualization ........................................... 1
1.2 Motivation .................................................. 2
1.3 Objectives .................................................. 3
1.4 Organization ............................................... 3

2 Background

2.1 Initial Remarks ............................................. 4
2.2 Systems-of-Systems ......................................... 5
   2.2.1 Definitions of Systems-of-Systems .................... 5
   2.2.2 Comparison with Monolithic Systems ................. 5
   2.2.3 Characterization of Systems-of-Systems ............. 6
   2.2.4 Classification of Systems-of-Systems ............... 8
   2.2.5 Systems-of-Systems Engineering ...................... 10
   2.2.6 Examples of Systems-of-Systems ....................... 11
2.3 Ontologies .................................................. 12
   2.3.1 Terminology and Basic Concepts ..................... 12
   2.3.2 Uses of Ontologies .................................. 13
   2.3.3 Classification of Ontologies ......................... 14
   2.3.4 Languages for Representing Ontologies ............. 15
   2.3.5 Support Tools ....................................... 19
   2.3.6 Examples of Ontologies ................................ 19
   2.3.7 Methodology ......................................... 21
2.4 Knowledge Representation for Systems-of-Systems ............ 24
2.5 Final Remarks ............................................... 25

3 OntoSoS: An Ontology for Systems-of-Systems ................. 26

3.1 Initial Remarks ............................................. 26
3.2 Methodology Activities .................................... 27
3.3 Final Remarks ............................................... 39
4 OntoSoS Evaluation 40
   4.1 Initial Remarks ........................................... 40
   4.2 Survey Structure .......................................... 40
   4.3 Results ...................................................... 47
   4.4 Final Remarks .............................................. 56

5 Conclusion 58
   5.1 Characterization and Contributions of the Research .......... 58
   5.2 Difficulties and Limitations .................................. 59
   5.3 Future Work ............................................... 59

References 65

Appendix A OntoSoS Specification 66

Appendix B Evaluation Survey 70
## List of Figures

| Figure | Title | Page |
|--------|-------|------|
| 2.1    | Scale of Characteristics Associated to SoS | 8    |
| 2.2    | Types of SoS                                 | 9    |
| 2.3    | Gruber Ontology Definition                   | 13   |
| 2.4    | Uses for Ontologies                          | 13   |
| 2.5    | Spectrum of Ontology Kinds                    | 15   |
| 2.6    | Classification According to Generality        | 16   |
| 2.7    | RDF Statement Components                      | 16   |
| 2.8    | An RDF Graph with Two Nodes (Subject and Object) and a Triple Connecting Them (Predicate) | 17   |
| 2.9    | Screenshot of Protégé Tool                    | 20   |
| 2.10   | Fragment of the Class Taxonomy of Cyc’s Ontology | 21   |
| 2.11   | Partial View of the Taxonomy of the Enterprise Ontology | 22   |
| 2.12   | Development Process and Life Cycle of METHONTOLOGY | 23   |
| 3.1    | OntoSoS - UML Class Diagram                  | 34   |
| 3.2    | Protégé - OntoSoS Classes                    | 36   |
| 3.3    | Protégé - OntoSoS Object Properties           | 37   |
| 3.4    | Protégé - OntoSoS Visualization with OWLViz   | 38   |
| 4.1    | OntoSoS - UML Class Diagram for Evaluation   | 41   |
| 4.2    | Researchers’ Opinions on Description of Concepts | 48   |
| 4.3    | Researchers’ Opinions on Ontology Coverage   | 48   |
| 4.4    | Researchers’ Opinions on Ontology Relationships | 49   |
| 4.5    | Researchers’ Opinions Regarding the Need of an Ontology | 49   |
| 4.6    | Researchers’ Will in Using OntoSoS on Their Next Projects | 50   |
List of Tables

| Table | Title                                                                 | Page |
|-------|----------------------------------------------------------------------|------|
| 3.1   | OntoSoS Glossary                                                    | 28   |
| 3.2   | OntoSoS Taxonomies                                                  | 32   |
| 3.3   | OntoSoS Relations                                                   | 33   |
| 4.1   | OntoSoS Glossary for Evaluation.                                    | 42   |
| 4.2   | Survey Questions                                                    | 46   |
| 4.3   | Evaluation Groups, Subgroups, and Codes                             | 50   |
| 4.4   | Researchers’ Suggestions                                            | 51   |
| 4.5   | Justification for Suggestions Not Accepted                          | 55   |
Introduction

1.1 Contextualization

Recently, there is a growing interest in a class of more complex systems, which result from the cooperation of other systems. Systems-of-Systems (SoS) are "supersystems", composed of elements that already represent independent systems, which interact among themselves to achieve a common goal (Jamshidi, 2008a). These systems can reach goals that they could not reach when operating on their own (Jamshidi, 2008b). SoS are present in many domains and can be found everywhere, for example (Lane, 2013): (i) within our homes, the "smart homes" integrate power systems, fire alarms, and communication systems; (ii) within companies, integrating employee systems, payroll systems, and accounting systems.

Despite being commonly used, there is no widely accepted definition for the term "Systems-of-Systems" (IEEE-Reliability-Society, 2014) (Maier, 1998). There are several definitions in the literature and there is no consensus among the authors. However, the notion on what these systems represent is generally recognized. This is a kind of system that is built from other systems that already represent large-scale systems themselves. The main characteristics of an SoS are operational independence, managerial independence, evolutionary and adaptive development, emergent behavior and geographic distribution (Jamshidi, 2005) (Maier, 1998) (Sage and Cuppan, 2001). Integrated air defense...
networks, the Internet, and enterprise information networks are examples of SoS (Maier, 1998).

The importance of the research field of SoS is emphasized by Jamshidi (2008a), who points out that there is a large gap from basic definitions up to theory, management and implementation. The emerging demand for systems with those characteristics is growing and it is important that the concepts, characteristics, architectural patterns and quality attributes are widespread among practitioners and researchers of the field (Jamshidi, 2008a). This will facilitate knowledge sharing and communication, contributing to the development of this research field.

Simultaneously, it has been observed an interest in proposing ontologies, which can be defined as an explicit specification of a conceptualization (Gruber, 1993). Ontologies can be developed to support communication, interoperability, and systems engineering (Uschold and Gruninger, 1996). An ontology establishes a common understanding by providing an accurate and unambiguous communication between people and organizations (Uschold, 1996). Ontologies have been used to represent knowledge in many fields, such as Systems Engineering (van Ruijven, 2013), Software Architecture (Babu et al., 2007), Software Testing (Souza et al., 2013) (Barbosa et al., 2006), and Software Reuse (Silva et al., 2014). In the same perspective, an ontology for SoS could contribute to the field by formalizing concepts and providing a common understanding between the SoS researchers and practitioners.

1.2 Motivation

The main problem that has motivated this Master’s project is that there is a lack of common understanding of the concepts and terms related to SoS. The emerging field of SoS is not consolidated yet and there is a lack of consensus in the terminology used (Jamshidi, 2008a). There are many challenges for stakeholders dealing with SoS, because there are several related terms and concepts that can vary a lot, leading to misunderstandings and misinterpretations. Due to this lack of clarity, it is necessary to properly represent and interpret SoS concepts and terms correctly and consistently (Barot et al., 2013).

Defining terms, establishing a common vocabulary, and formalizing concepts related to SoS will help to solve the theoretical problems regarding standards of SoS, identified by Jamshidi (2008a). In this perspective, an ontology for establishing a common understanding in the SoS field is required.
CHAPTER 1. INTRODUCTION

1.3 Objectives

This Master’s project had as its main goal to establish OntoSoS, an ontology that formally describes the terms and concepts related to SoS. Consequently, a common vocabulary was also created, allowing the common understanding and knowledge sharing in the SoS community.

The ontology was built based on METHONTOLOGY (Fernández-López et al., 1997), a methodology for building ontologies. The main activities of this methodology are specification, conceptualization, formalization, implementation, and maintenance. The activities for specification and conceptualization of the ontology were conducted in the context of the research group in which this project is part of. Therefore, such activities had support of researchers in the SoS field. The formalization and implementation activities were conducted using Protégé tool\(^1\). To evaluate the coverage and correctness of the ontology, a complementary activity was conducted. The proposed ontology was submitted to researchers, who were able to suggest refinements and corrections.

As a result, OntoSoS makes it possible to reuse knowledge of the SoS field, supporting the learning and spreading terms and concepts related to SoS. As a consequence, this Master’s project intends to contribute to the consolidation of the SoS field.

1.4 Organization

This dissertation is organized as follows: Chapter 2 presents the necessary background on SoS and ontologies. It includes the terminology and basic concepts of SoS, including a comparison to traditional systems, characterization, classification and engineering of this class of systems, and presents examples of SoS. Moreover, it focuses on definitions and uses of ontologies, classifications, languages, methodologies, and examples of some of the most outstanding ontologies.

Chapter 3 presents OntoSoS: an Ontology for SoS. The methodology for modelling and building OntoSoS is presented, with all steps and activities followed to build it, including the artifacts generated and also the implementation process.

Chapter 4 presents the results of an evaluation of OntoSoS conducted with researchers of the SoS field. The version of the ontology that was used in the evaluation is presented, and suggestions and improvements provided by the researchers are also described.

Chapter 5 presents the conclusions of this dissertation and provides ideas for future work, proposing future approaches and uses for OntoSoS.

\(^1\)http://protege.stanford.edu/
2.1 Initial Remarks

SoS are a special type of system which are composed by other systems, named constituent systems. By working alone, these constituent systems, also known as monolithic systems, cannot accomplish the objectives that they obtain when working together.

Different definitions can be found in the literature for SoS focusing on different characteristics. Some of them emphasize the fact that the constituent systems are collaboratively integrated and have two additional properties: operational and managerial independence (Maier, 1998). Others highlight that SoS exist when the constituent systems work connected and obtain results that are unachievable by them individually (Kriengle, 1999).

In this context, ontologies can be used to define SoS, having the terms listed as classes and relationships that represent the definition. Ontologies define the terms used to describe and represent the knowledge of a given area (W3C, 2004a), providing a shared understanding (Gruber, 1993). Basically, an ontology contains classes (things), relationships among these classes and their properties (attributes) (W3C, 2004a).

In this chapter basic concepts regarding SoS are presented. First, some definitions are presented, then a comparison with monolithic systems is done. Next, the characterization of SoS is presented. Afterwards, the common classification for SoS is presented. SoS engineering is discussed and then some examples of this kind of system are presented.
Moreover, basic concepts and terminology concerning ontologies are presented, as well as the uses for ontologies and kinds of problems they can contribute to solve. Classification for ontologies and languages for representing ontologies are also discussed. Finally, examples of ontologies and the methodology are presented.

## 2.2 Systems-of-Systems

### 2.2.1 Definitions of Systems-of-Systems

An SoS is a kind of systems composed by other systems. Some definitions for SoS have been proposed in literature. Here some of them are presented:

- “An SoS is a set of collaboratively integrated systems that possess two additional properties: operational independence of the components and managerial independence of the components” (Maier, 1998).

- “An SoS is a set of different systems so connected or related as to produce results unachievable by the individual systems alone” (Kriengle, 1999).

- “SoS exist when there is a presence of a majority of the following five characteristics: operational and managerial independence, geographic distribution, emergent behavior, and evolutionary development” (Sage and Cuppan, 2001).

- “SoS can be loosely defined as a combination of systems in order to fulfill some kind of capability, with the additional fact that the composing systems should have operational and managerial independence” (Autran et al., 2008).

- “SoS are large-scale integrated systems that are heterogeneous and independently operable on their own, but are networked together for a common goal” (Jamshidi, 2008a).

- “An SoS is any system that is relatively large and complex, dynamically evolving, and physically distributed system of pre-existing, heterogeneous, autonomous, and independently governed systems, whereby the system of systems exhibits significant amounts of unexpected emergent behavior and characteristics” (Firesmith, 2010).

### 2.2.2 Comparison with Monolithic Systems

SoS differ from monolithic systems in some aspects (Oberndorf and Sledge, 2010). In an SoS the causes of problems and effects of behaviors are a combination of factors, which may be known or unknown. The dependencies are often largely outside a single
program’s span of control and the context may not be completely known by its engineers and managers. In a monolithic system, the dependencies are within the system itself, and it is less difficult, for instance, to estimate the impact of change requests in the system. Regarding the goals of the system, in an SoS they refer to the capabilities of the constituent systems plus the emergent capabilities of the SoS.

Moreover, the focus of the SoS must satisfy, suffice, and comprise to achieve the collective emergent capabilities, and not only the constituent systems’ features. With respect to the negotiations and decisions, in an SoS there is more dependence on collaboration and influence at best, but sometimes, when negotiations are unsuccessful, mitigation may be the only way to deal with some problems.

These are common issues often found in SoS projects from different domains, such as defense integrated networks, airport systems and smart-systems, which include smart grids, smart buildings and smart cities (Maier, 1998). Since there is an emergence for this kind of system, and based on the aforementioned differences, it is notable that SoS require new paradigms of working so practitioners can deal with their characteristics.

2.2.3 Characterization of Systems-of-Systems

A System of System can be defined by the presence of a majority of the five following characteristics: operational and managerial independence, geographic distribution, emergent behavior, and evolutionary and adaptive development (Jamshidi, 2005) (Maier, 1998) (Sage and Cuppan, 2001). Each characteristic is shortly discussed below.

- **Operational independence**: if an SoS is disassembled, its component systems (the constituents) must operate independently.

- **Managerial independence**: the component systems maintain operational existence independent of the SoS.

- **Evolutionary development**: the SoS keeps continually being evolved, that is, new features can be added, changed or removed, according to new requirements. This characteristic may also result in a dynamic architecture, where constituents can be incorporated or removed while the SoS is running (Nakagawa et al., 2015).

- **Emergent behavior**: the SoS function is not placed in any constituent system, but belongs to the SoS as a whole.

- **Geographic distribution**: the component systems may be in different locations.

Firesmith (2010) divides common Systems-of-Systems characteristics as system-level, which describe the SoS as a whole (complexity, evolution, negative emergence, size, and
variability), and subsystem-level, which describes the SoS in terms of characteristics of its subsystems (autonomy, governance, heterogeneity, physical distribution, and reuse).

The system-level characteristics are the following (Firesmith, 2010):

- **Complexity**: the degree to which a system is difficult for its stakeholders to understand and analyze, especially due to having a large number of components connected by many complicated interfaces.

- **Evolution**: the degree to which the goals and requirements for a system (and its subsystems) change over time.

- **Negative emergence**: the degree to which the new behaviors and characteristics of a system that result (emerge) from the interaction of the system’s subsystems are detrimental, unintended, and difficult to predict from the behaviors and characteristics of these individual subsystems.

- **Size**: the amount or magnitude of the system with regard to a suitable dimension (for example, the number of constituents, the amount of software - lines of code - in the system and so on).

- **Variability**: the degree to which a single type of system simultaneously exist in multiple variants, versions, or configurations.

The subsystem-level characteristics are the following (Firesmith, 2010):

- **Autonomy**: the degree to which the subsystems within a system are independent, stand alone and are individually useful, self-contained, and operationally independent (neither controlled by nor controlling other subsystems).

- **Governance**: the degree to which the subsystems of a system are governed (specified, managed, funded, developed, owned, operated, maintained, and sustained) in an independent, decentralized, and uncoordinated manner.

- **Heterogeneity**: the degree to which the subsystems of a system differ each other in that they have different goals, objectives, and requirements; have different behavior and characteristics; provide unrelated functionality; belong to different application domains; and are implemented using different technologies.

- **Physical distribution**: the degree to which the subsystems of a system exist in different physical locations.
• **Reuse**: the degree to which the subsystems of the system have been reused regardless as to whether they are commercial-off-the-shelf (COTS), government-off-the-shelf (GOTS), military-off-the-shelf (MOTS), organizational-internal reuse, open source, and freeware.

Such characteristics are commonly used into various existing definitions of Systems-of-Systems in the literature, and different definitions emphasize different characteristics. Each characteristic can be associated to a scale, in which common systems (“trivial”) have lower degrees, and SoS (“ultra-large scale systems”) have greater degrees (Firesmith, 2010). The scales and characteristics are presented in Figure 2.1.

![Figure 2.1: Scale of Characteristics Associated to SoS (Firesmith, 2010)](image)

### 2.2.4 Classification of Systems-of-Systems

Based on management approaches and on the relationships among the systems in the SoS, different types of SoS can exist, as illustrated in Figure 2.2. Maier (1998) identified three types of SoS: *virtual*, *collaborative*, and *directed*. Then, a new type, *acknowledged*, has been proposed (Dahmann and Baldwin, 2008):
• **Virtual:** This type of SoS lack a central management authority. The SoS also lack a centrally agreed purpose. The component systems may not be recognized.

• **Collaborative:** The central management organization does not have coercive power to run the system. The component systems voluntarily collaborate to fulfill the central purposes. There is no SoS engineering team to guide or manage the activities related to SoS in its constituents.

• **Acknowledged:** This type of SoS has recognized objectives, a designated manager and resources for the SoS. However, the constituent systems keep their independence, objectives, funding, and development approaches. Changes in the systems are based on collaboration between the SoS and the system.

• **Directed:** In this type, the integrated SoS is built and managed to fulfill specific purposes. The component systems operate independently, but their normal operation is subordinated to the central managed purpose. The evolution of the SoS is controlled. For instance, an integrated air defense network is centrally managed to defend a region against enemies, but its systems may operate independently.

![Figure 2.2: Types of SoS](Lane, 2013)


2.2.5 Systems-of-Systems Engineering

New paradigms are required to deal with differences in SoS when compared to conventional systems. The processes, artifacts, and collaborations necessary for engineering a SoS are dynamic and there is a constant evolution (Oberndorf and Sledge, 2010).

The United States Department of Defense (DoD) mentions that Systems-of-Systems Engineering (SoSE) “deals with planning, analyzing, organizing, and integrating the capabilities of a mix of existing and new systems into an SoS capability greater than the sum of the capabilities of the constituent parts” (DoD, 2008).

The DoD argues that there are seven core elements of SoSE that provide the context for the application of systems engineering processes in an SoS environment. The core elements are described below (DoD, 2008):

1. **Translating capability objectives.** The objectives of a SoS are typically conceived as capabilities. Therefore, stakeholders (systems engineer, SoS manager, and users) need to work together so these capabilities can be translated into SoS high-level requirements, which can provide the creation for the technical planning to evolve the capability over time.

2. **Understanding systems and relationships.** Understanding the systems involved in the SoS and their relationship and interdependencies is one of the most important aspects of SoSE. Understanding the functionality of each constituent system in the SoS will help to understand (i) how the systems support the SoS objectives, (ii) technical details, and (iii) SoS development plans, considering timing and synchronization aspects. In addition, the SoSE team needs to identify the stakeholders of both SoS and its systems constituents, so it is possible to understand their organizational context and their role in the SoS over time.

3. **Assessing performance to capability objectives.** The SoSE team needs to define metrics and methods for assessing performance of the SoS capabilities, which can be established by identifying the most important mission threads and focusing the assessment effort on end-to-end performance. The performance may be monitored based on operational experience and issues.

4. **Developing and evolving an SoS architecture.** Functions, relationships, and dependencies of constituent systems are included in the architecture of an SoS. “The architecture of the SoS provides the technical framework for assessing changes needed in systems or other options for addressing requirements”. Since the systems contributing the the SoS typically exist when the SoS is conceived, the SoSE
team needs to consider the current state and plans of the constituent systems when developing an architecture for the SoS.

5. **Monitoring and assessing changes.** The changes that will impact SoS functionality or performance can be internal changes in the constituent systems or external demands on the SoS. The SoSE team must be aware that the constituent systems evolve independently and this could affect the SoS.

6. **Addressing requirements and solution options.** The requirements can be at SoS or constituent systems level, so “a process is needed to collect, assess, and prioritize user needs, and then evaluate options for addressing these needs”. The SoSE team needs to work with requirements managers of the constituent systems to identify the specific requirements and to which systems they have to be addressed.

7. **Orchestrating upgrades to SoS.** The SoSE team needs to work with sponsors, managers, system engineers, and contractors in order to orchestrate all the process of upgrade, leading the coordination, integration, and tests across the SoS to ensure that the changes are implemented correctly.

### 2.2.6 Examples of Systems-of-Systems

Some good examples of SoS are presented in the study of Maier (1998). An Integrated Air Defense is composed of geographically distributed network of semiautonomous elements. These elements can be radars, surveillance systems, missile launch batteries, missile tracking and control sites, fighter aircraft, and so on. When operating together, the whole system shows emergent behavior. This is an example of a directed SoS.

The Internet is an example of collaborative SoS. Its elements are computers networks and major computer sites. Internet components collaboratively exchange information using documented protocols. The development, management, and operation are a collaborative effort.

Another good example of a SoS is the Global Earth Observation System-of-Systems (GEOSS) (Butterfield et al., 2008). “It is comprised of operational and research systems that are created for their own purposes, usually to satisfy national or regional requirements.”. GEOSS is a global public infrastructure which must generate environmental data and analyses. The purpose of this SoS is to interconnect the existing and future observation systems (for instance, floating buoys monitoring the oceans’ temperature and salinity, meteorological stations, radar systems and so on). It also seeks interoperability of all these systems and aims to reduce costs and promote international cooperation (Luzeaux and Ruault, 2010). GEOSS aims to:
• reduce the loss of property and human lives resulting from natural or human-induced disasters;
• understand the environmental factors which impact health and well-being;
• improve the management of energy resources;
• understand, assess, predict, mitigate and adapt to climatic changes and variability;
• improve the management of freshwater resources through a better understanding of the water cycle;
• improve the weather information, forecast and warning;
• improve the management and the protection of the terrestrial, coastal and marine ecosystems;
• encourage sustainable agriculture and fight against desertification;
• understand, monitor and preserve biodiversity.

2.3 Ontologies

2.3.1 Terminology and Basic Concepts

In Computer Science, ontologies can be used to represent knowledge. The W3C (2004a) (World Wide Web Consortium) defines an ontology as a set of terms used to describe and represent an area of knowledge. According to Gruber (1993), an ontology is defined as an explicit specification of a conceptualization. This definition is represented in Figure 2.3. Later, Studer et al. (1998) extended Gruber’s definition by proposing that an ontology is a formal explicit specification of a shared conceptualization. In their definition, conceptualization refers to an abstract model of something in the world, in which its relevant concepts are identified. Explicit means that the types of concepts, constraints and usage are explicitly defined. Formal means that the ontology should be machine-readable, that is, it should exclude natural languages. Lastly, shared means that the consensual knowledge captured by the ontology must be accepted by a group, that is, it cannot be private to some individual.


2.3.2 Uses of Ontologies

Regarding their space of use (purpose), ontologies can be used for communication between people, interoperability among systems, or systems engineering (Uschold and Gruninger, 1996) (Uschold, 1996). These uses for ontologies are summarized in Figure 2.4.

Communication refers to sharing a common understanding of concepts, thus providing a common vocabulary of the terms, their meaning and relationship for people with differ-
ent needs and viewpoints in a given context. Here ontologies can facilitate communication among people. This way, ontologies can be used as: (i) normative models, when different people must have a shared understanding of a system, (ii) networks of relationships, providing logical connections between elements, (iii) consistency and lack of ambiguity, providing definitions for terms, and (iv) integrating different user perspectives.

Interoperability refers to the data exchange among systems that need to interpret concepts using different software tools. For example, ontologies can be used to support translation between different languages and representations.

Regarding systems engineering, ontologies can be used to support activities in the design and development of software systems. They can provide reusability, when an ontology is the basis for formal encoding of important entities, attributes, processes and their relationships. Such representation may be a reusable or a shared software component. Ontologies can also guide the knowledge acquisition when building knowledge-based systems. Also, a formal representation makes it possible the automation of consistency in a system, resulting in a more reliable software. Finally, an ontology can assist in identifying requirements and defining a specification for a system.

2.3.3 Classification of Ontologies

Some classifications for ontologies have been proposed in literature. They are presented in this section.

Internal Structure

Ontologies can be classified according their degree of formality (Uschold, 1996), that is, its internal structure. An informal ontology is expressed in natural language or some restricted and structured form of a natural language, such as glossaries or controlled vocabularies. A semi-formal ontology is expressed in an artificial formally defined language, such as conceptual models or UML diagrams (UML, 2014). Formal ontologies define terms with formal semantics, including first order logic and axioms, description logics or some machine-readable language, such as OWL (W3C, 2014c) or RDF (Resource Description Framework) (W3C, 2004b).

Figure 2.5 shows the spectrum of ontology kinds based on their internal structure and degree of formality. Lightweight ontologies make no use of axioms, whereas heavyweight ontologies make intensive use of axioms for specification (Wong et al., 2012).

In the spectrum of ontology kinds, Terms can be seen as a controlled vocabulary, that is, a finite list of terms belonging to a given area. Glossaries are lists of terms with their meanings specified as natural language statements. A taxonomy is a controlled
vocabulary organized into a hierarchical or parent-child structure. Thesauri are similar to a taxonomy, with the addition of relationships, such as associative or equivalence. Frames include classes and their properties. The more the ontology is placed to the right side of the figure, the more expressiveness and formality it provides.

**Generality**

An ontology may also be classified according to its generality. This classification was proposed by Guarino (1998) and it is presented in Figure 2.6. Top-level ontologies describe generic concepts, such as space, time and events. They are domain-independent and could be reused when building new ontologies. Domain ontologies describe the vocabulary relative to a specific domain by specializing concepts that are in the top-level ontology. Task ontologies describe the vocabulary concerning a generic task or activity by specializing concepts that are in the top-level ontology. Application ontologies are more specific, its concepts are, in general, roles played by entities of the domain when performing a given task.

**2.3.4 Languages for Representing Ontologies**

Many languages for representing ontologies were proposed in the 90s. Languages were created based on Artificial Intelligence, mostly based on first order logic, such as KIF (Knowledge Interchange Format). However, the boom of Internet led to the creation of ontology languages based on the Web, and their syntax are based on XML (Breitman, 2005) (Gómez-Pérez et al., 2004).
CHAPTER 2. BACKGROUND

Figure 2.6: Classification According to Generality (Guarino, 1998)

**RDF and RDFS**

The RDF (Resource Description Framework) (W3C, 2004b) is a standard model for data interchange on the Web. RDF is represented by sentences, containing a subject (resource), a predicate (property), and an object, which can be a data value or a resource, as can be seen in Figure 2.7.

| RDF Statement |
|----------------|
| Subject | Predicate | Object |
| Resource | Property | Data Value or Resource |

Figure 2.7: RDF Statement Components (Lacy, 2005)
An RDF statement can be also represented by a graph, which is useful for communication among humans. This representation is presented in the Figure 2.8.

**Figure 2.8:** An RDF Graph with Two Nodes (Subject and Object) and a Triple Connecting Them (Predicate)  
(W3C, 2014a)

The machine-readable RDF representation is based on the XML syntax. Some of the elements are `rdf:RDF` and `rdf:Description`, which are useful for representing metadata, but not enough for representing ontologies. Due to that, the RDF-Schema (W3C, 2014b) was created. Basically, the RDF-Schema is a language for describing vocabularies with properties and classes. The RDF-Schema provides a limited number of pre-defined elements, and new classes and properties can be described. In this context, RDF-Schema is a framework for describing classes and properties. Some of the main elements of RDF-Schema are `rdfs:Resource`, `rdfs:Class`, `rdfs:Literal`, `rdfs:Property`, `rdfs:Statement`, `rdfs:subClassOf`, `rdfs:subPropertyOf`, `rdfs:type`, and `rdfs:seeAlso`.

The RDF and RDF-Schema are used together and the language using both is referred as RDFS. The RDFS offers a set of elements for modeling simple ontologies and does not provide logic constructs, like disjunction.

**SHOE**

The language SHOE (Simple HTML Ontology Extension) (Luke and Heflin, 2000) was developed at the University of Maryland. It was created as an extension of HTML, aiming to incorporate machine-readable knowledge in Web documents, providing specific tags for representing ontologies (Gómez-Pérez et al., 2004). The main goal of SHOE was to provide tags for making relevant information of web pages content available, allowing more precision on the search engines. SHOE allows definitions of concepts, relationships and attributes. It presents less expressivity than RDF and also it is difficult to provide maintenance of annotated web pages. This language has been discontinued and the researchers adopted DAML + OIL and OWL (Breitman, 2005).

**OIL**

The language OIL (Ontology Inference Layer) (Fensel et al., 2001) was created in the context of the project On-To-Knowledge and it was built to express semantics of resources because RDF does not provide the necessary formalism to support inference mechanisms.
CHAPTER 2. BACKGROUND

The semantics of and the inference mechanism is based on description logic. OIL encompasses description logic (formal semantics and inference support), frame-based systems (for modeling) and web languages (based on XML and RDF).

An OIL ontology is organized in three levels: the ontology container (information about the ontology), the ontology definition (ontology terms definitions), and the object level (instances) (Gómez-Pérez et al., 2004) (Breitman, 2005).

**DAML**

At the same time OIL was being created, the DARPA (Defense Advanced Research Projects Agency) was developing the language DAML (DARPA Agent Markup Language) (DARPA, 2000), extending RDF by adding more expressiveness. The aim of this language was to make it easier interaction among software agents. Since DAML has inherited many aspects of OIL, the features of both languages are quite similar (Breitman, 2005).

**DAML + OIL**

The language DAML + OIL (van Harmelen et al., 2001) replaced the languages DAML and OIL by combining both. DAML + OIL ontologies are written in XML and are based on RDFS. The language is divided in objects domain (class members of the ontology) and datatypes domain (values imported from XML). DAML + OIL is composed by class elements, class expressions and properties (Breitman, 2005). The class elements associate a class to its definition, including RDFS and DAML elements. The class expressions are possible ways of referencing a class (by name, enumeration, boolean combinations and restriction). The properties associate a property to its definition, also including RDFS and DAML elements.

**OWL**

The language OWL (Web Ontology Language) (W3C, 2014c) is the current standard for coding ontologies and it is a W3C recommendation. It is a revision of DAML + OIL and was designed to meet the needs of the Semantic Web.

According to the W3C (2014c), “OWL is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be exploited by computer programs, e.g., to verify the consistency of that knowledge or to make implicit knowledge explicit. OWL documents, known as ontologies,
can be published in the World Wide Web and may refer to or be referred from other OWL ontologies”.

The current version of OWL is the OWL 2, which is a revision and extension of the first version. OWL 2 ontologies provide classes, properties, individuals, and data values, and can be used along with information written in RDF. OWL 2 has three different profiles (W3C, 2012b): OWL 2 EL, OWL 2 QL, and OWL 2 RL.

OWL 2 EL is useful in applications employing ontologies that contain very large numbers of properties and/or classes. This profile captures the expressive power used by many such ontologies and is a subset of OWL 2.

OWL 2 QL is aimed at applications that use very large volumes of instance data, and where query answering is the most important reasoning task.

OWL 2 RL is aimed at applications that require scalable reasoning without sacrificing too much expressive power. It is designed to accommodate OWL 2 applications that can trade the full expressivity of the language for efficiency, as well as RDF(S) applications that need some added expressivity.

2.3.5 Support Tools

Building ontologies is a complex and time-consuming activity. If developers had to implement ontologies directly by writing the ontology language without any support, it would be much more complex. To make it easier, environments for building ontologies have been created.

Protégé

Protégé is an open source ontology editor created by the Stanford University. It is a tool that enables the construction of ontologies, allowing the definition of classes, class hierarchies, variables, variable-value restrictions, and the relationships between classes and the properties of these relationships. Protégé comes with visualization packages, such as OntoViz, which helps the user to visualize ontologies with diagrams (Kapoor and Sharma, 2010). In Figure 2.9 an screenshot of Protégé is presented.

2.3.6 Examples of Ontologies

In this section some known ontologies are presented as examples.
CHAPTER 2. BACKGROUND

Cyc

The Cyc\(^1\) project (enCYClopaedia) had a huge effort in developing a top-level ontology. This ontology was designed to gather human knowledge. The upper classes of Cyc are organized in categories, such as time, dates, and space. The root of the ontology is the class \textit{Thing}. The Cyc ontology is implemented in the CycL language.

The Cyc ontology can be used to support, for instance, the comprehension of natural language or the sharing of knowledge within different groups.

Cyc is proprietary and restricted. However, a free version of Cyc, the OpenCyc, is available at Cycorp website. Figure 2.10 presents a fragment of the Cyc Ontology.

WordNet

WordNet\(^2\) is a lexicon database developed at Princeton University that provides meanings for the English language and the main relation among words is synonymy. It is a linguistic

\(^1\)http://www.cyc.com/
\(^2\)http://wordnet.princeton.edu/
ontology and its purpose is to describe semantic constructs instead of model a specific area. WordNet divides the lexicon into five categories: nouns, verbs, adjectives, adverbs, and function words.

The Enterprise Ontology

The Enterprise Ontology\(^3\) (Uschold et al., 1998) is a domain ontology developed within the Enterprise Project at the University of Edinburgh, in a partnership with some companies, such as IBM. The ontology contains terms and definitions relevant to the business domain. Figure 2.11 presents a fragment of the Enterprise Ontology.

2.3.7 Methodology

METHONTOLOGY was developed in the Ontology Group at Universidad Politécnica de Madrid. It is based in the main activities identified in the software development process of IEEE and in knowledge engineering methodologies. This methodology enables the construction of ontologies at knowledge level, that is, the conceptual level, as opposed to the implementation level. In METHONTOLOGY, the ontology development process is concerned with which activities have to be performed when building ontologies. The methodology specifies three groups of activities (Gómez-Pérez et al., 2004):

\(^3\)http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html
CHAPTER 2. BACKGROUND

Figure 2.11: Partial View of the Taxonomy of the Enterprise Ontology  
(Gómez-Pérez et al., 2004)

- **Ontology management activities:** include scheduling (planification), control, and quality assurance / control. *Planification* activity identifies the tasks that will be performed, their arrangement, and estimation of number of hours and resources. The *control* activity concerns about the completion of schedule tasks. The *quality control* activity assures that the quality of each output is satisfactory.

- **Ontology development-oriented activities:** include specification, conceptualization, formalization, implementation, and maintenance. The *specification* activity states why the ontology is being build, its purpose, and who are the end-users. The *conceptualization* activity defines which terms will be in the ontology and structures the knowledge as conceptual models. The *formalization* activity turns the conceptual models into a formal or semi-computable model. The *implementation* activity builds the models in an ontology language. The *maintenance* activity updates and refines the ontology if needed.

- **Ontology support activities:** this group includes a series of activities performed at the same time as the development activities. It includes knowledge acquisition, integration, evaluation, documentation and configuration management. *Knowledge acquisition* aims to acquire knowledge from experts in the area. The *integration* activity is performed when building a new ontology by reusing other ontologies already available. The *evaluation* activity makes a technical judgement of the ontology, of the software environments, and of the documentation. The *documentation* activity details each and every completed stage. The *configuration management* activity records all the versions of the documentation and the ontology code so all changes can be controlled.

In Figure 2.12 all activities of METHONTOLOGY are presented, as well as their arrangement and order of execution. As it can be seen in this figure, in the support activities, the knowledge acquisition, integration, and evaluation are greater during ontology
conceptualization and they decrease during formalization and implementation. This happens because (i) most of the knowledge is acquired at the beginning of the process, (ii) the integration with other ontologies is not postponed to the implementation, and (iii) the conceptualization must be evaluated accurately to avoid propagating errors.

Figure 2.12: Development Process and Life Cycle of METHONTOLOGY (Corcho et al., 2005)
2.4 Knowledge Representation for Systems-of-Systems

As SoS may involve several disciplines, specialists, and technologies, an adequate communication is essential. However, stakeholders of SoS can face several difficulties by using related terms and concepts with different meanings and purposes.

Knowledge Representation is a subarea of Artificial Intelligence concerned with understanding, designing, and implementing ways of representing information so that computers can use it (Shapiro, 2006). Knowledge Representation approaches, such as ontologies, vocabularies, taxonomies, and thesauri, could offer an important support for disseminating a shared understanding of SoS terms and concepts.

Aiming to characterize the application of Knowledge Representation approaches to the field of SoS, a Systematic Literature Review (SLR) (Kitchenham and Charters, 2007) has been conducted (Abdalla et al., 2015a) (Abdalla et al., 2015b). Besides investigating which and how Knowledge Representation approaches have been applied to SoS, an investigation was also done to identify in which context they have been applied and the potential benefits that they could bring to the development of SoS.

The main goals of the SLR conducted was to identify:

- Knowledge Representation approaches applied to SoS and their degree of formality.
- Motivations for using such approaches and their space of use.
- The application domains the approaches were applied to and if they were applied to a real case study or system.
- Terms covered by approaches in SoS.

Results showed that the most used approach in the SoS field is formal ontology and interoperability is the most addressed space of use. Moreover, there is a relation between interoperability and the degree of formality. Since the interoperability between systems requires machine-readable approaches, the studies addressing this space of use are more likely to use formal approaches. On the other hand, studies that address SoSE as the main space of use tend to use semi-formal or informal approaches due to the fact that they are used to guide systems engineering or knowledge acquisition.

The degree of formality is also related to the validation of the approaches proposed in studies. In this perspective, formal approaches tend to be validated, either using toy scenarios or with industrial cases.

Many studies analyzed in the SLR did not address a specific application domain. Although the “general” application domain was pointed as predominant, there was no consistency among the extracted terms. No direct relationship among the terms extracted
from the approaches could be identified. Moreover, there were not so many repeated terms across the studies. For instance, the term stakeholder, which could be related to the communication space of use, was repeated across only three different studies. It suggests that there is a lack of consistency on the terminology of the studies. A reason for this lack of consistency could be that the studies found addressed specific tasks, such as crisis management and requirements engineering, and did not concern with the SoS field as a whole. In studies addressing interoperability, for example, the approaches contained terms specifically related to a given context. Hence, a common understanding could not be established from those terms, which might be a consequence of the lack of consensus in SoS definitions.

In this sense, an approach to formally define and relate the concepts and terms in the SoS field could contribute by establishing a common understanding, also supporting the communication among the SoS community.

\section*{2.5 Final Remarks}

In this chapter basic concepts related to SoS have been presented. First, some definitions for SoS were presented and then a comparison with monolithic systems was done. After that, a characterization of SoS was presented. Next, the classification for SoS was presented, discussing each type. Engineering of SoS was discussed and finally examples of SoS were presented.

A background of ontologies was introduced, which is necessary for the development of this Master’s project. First, terminology and basic concepts were presented, as well as uses of ontologies, and their classifications. Some of the most known languages for implementing ontologies were introduced, and then METHONTOLOGY was presented. Finally, the SLR conducted on Knowledge Representation for SoS were presented.

In the next chapter the process of modeling and implementation of OntoSoS is presented.
OntoSoS: An Ontology for Systems-of-Systems

3.1 Initial Remarks

In order to build an ontology it is important to follow a proper methodology that guides the process and helps in modelling and implementing an ontology. The aim of a methodology is to capture concepts that will be present in the ontology. In the ontology modeling, the main concepts and terms are identified, as well as definitions, classifications, and relationship among them. Thus, it was built a common vocabulary with the most relevant terms for SoS. These terms were collected from publications and researchers’ opinion.

In the context of this Master’s project, METHONTOLOGY (Fernández-López et al., 1997) was adopted as the methodology for creating the ontology. METHONTOLOGY is a mature methodology for building ontologies, and existing standards for traditional software development can be used as guidelines when adopting this methodology (Fernández-López et al., 2002). This methodology is divided into three main groups of activities: (i) management activities, (ii) development activities, and (iii) support activities. The development (technical) activities are the main flow of the methodology. The management and support activities occur in parallel, and were carried out during the development life cycle of OntoSoS. For this project, the development activities of speci-
fication, conceptualization and formalization are part of the ontology modeling. The development activity of implementation corresponds to the ontology implementation.

In this chapter, the process of modeling and implementing OntoSoS is presented, as well as the final version of the ontology.

3.2 Methodology Activities

The activities of METHONTOMETRY were conducted during the OntoSoS creation and are presented in this section.

Management Activities

Planification

An estimation of effort and arrangement of the schedule activities for this Master’s project were done. All activities involved in the ontology modelling and implementation were planned.

Control

Regular meetings with the advisor and the research group\(^1\) were conducted to follow-up the work being done and to guarantee the completion of scheduled activities.

Quality Control

Each activity output was validated by the advisor and by the research group. This ensured that the quality of outputs are satisfactory.

Development Activities

Specification

The specification of OntoSoS is presented as the following:

- **Name**: OntoSoS
- **Research Field**: System-of-Systems (SoS)
- **Purpose**: the purpose of OntoSoS is to establish a common understanding of the SoS field, facilitating the knowledge sharing and contributing to the evolution and consolidation of the field.

\(^{1}\)http://start.icmc.usp.br/
• **End-Users**: the end-users of OntoSoS are researchers, practitioners, and students with background in Software Engineering.

The specification is documented in the specification document, found in Appendix A.

**Conceptualization**

In this activity, the terms (concepts) of the ontology were defined and the knowledge was structured. All items presented here belong to the final version, after being evaluated in the ontology evaluation task, presented in Chapter 4.

The first task in the conceptualization phase is to define the list of terms that will be part of the ontology. The list of terms (also called glossary) was built based on different sources of knowledge: publications and researchers’ opinion. After the first list was created, brainstorming meetings were conducted within the research group to pre-validate it. Many inputs were given by researchers, and the list was continuously refined in meetings until it reached its first version. Along with the glossary, the definition of each term was also provided. In Table 3.1 the glossary with definitions is presented. The description of terms was based on the references mentioned. Some terms have been proposed by researchers during the conceptualization and there is no reference mentioned.

Table 3.1: OntoSoS Glossary

| Term                          | Description                                                                 | Reference |
|-------------------------------|-----------------------------------------------------------------------------|-----------|
| System-of-Systems             | A set or arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities. | (DoD, 2008) |
| Constituent                   | Each independent system that composes the SoS.                               | (Firesmith, 2010) (DoD, 2008) |
| Central Authority             | Entity coordinating the collaboration among constituents towards accomplishing a given goal. | (Maier, 1998) |
| Mission                       | A functional goal assigned to a system.                                      | (Silva et al., 2016) |
| Individual Mission            | Mission assigned to the constituent systems.                                 | (Silva et al., 2016) |
| Global Mission                | Mission assigned to the SoS.                                                 | (Silva et al., 2016) |
| System-of-Systems Characteristic | Key aspect related to an SoS.                                             | (Firesmith, 2010) |
| Constituent Characteristic    | Characteristics that primarily describe the system (SoS) in terms of the characteristics of its subsystems (constituents). | (Firesmith, 2010) |
| Complexity                  | The degree to which a system is difficult for its stakeholders to understand and analyze, especially due to having a large number of components connected by many complicated interfaces. | (Firesmith, 2010) |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| Evolutionary Development    | The degree to which (in terms of rate and impact) the goals and requirements for a system (and its subsystems) change over time.                                         | (Firesmith, 2010) |
| (Maier, 1998)               |                                                                                                                                                                             |                  |
| Size                        | The amount or magnitude of the system with regard to a suitable dimension.                                                                                                  | (Firesmith, 2010) |
| Variability                 | The degree to which a single type of system simultaneously exists in multiple variants, versions, or configurations.                                                          | (Firesmith, 2010) |
| Emergent Behavior           | The degree to which the new behaviors and characteristics of a system that result (i.e., emerge) from the interaction of the system’s subsystems are detrimental, unintended, and difficult to predict from the behaviors and characteristics of these individual subsystems. | (Firesmith, 2010) |
| (Maier, 1998)               |                                                                                                                                                                             |                  |
| Operational Independence    | The degree to which the subsystems within a system are independent, stand alone and are individually useful, self-contained, and operationally independent (i.e., neither controlled by nor controlling other subsystems). | (Firesmith, 2010) |
| (Maier, 1998)               |                                                                                                                                                                             |                  |
| Managerial Independence     | The degree to which the subsystems of a system are governed (e.g., specified, managed, funded, developed, owned, operated, maintained, and sustained) in an independent, decentralized, and uncoordinated manner. | (Firesmith, 2010) |
| (Maier, 1998)               |                                                                                                                                                                             |                  |
| **Heterogeneity** | The degree to which the subsystems of a system differ from each other in that they (1) have different goals, objectives, and requirements, (2) have different behavior and characteristics, (3) provide unrelated functionality, (4) belong to different application domains, and (5) are implemented using different technologies. | (Firesmith, 2010) |
| **Reuse** | The degree to which the subsystems of the system have been reused regardless as to whether they are commercial-off-the-shelf (COTS), government-off-the-shelf (GOTS), military-off-the-shelf (MOTS), organizational-internal reuse, open source, and freeware. | (Firesmith, 2010) |
| **Physical Distribution** | The degree to which the subsystems of a system exist in different physical locations. | (Firesmith, 2010) |
| **Dynamic Architecture** | It refers to architectures that change at runtime; i.e., their overall structure can be modified by adding, replacing, or withdrawing constituents, and even by changing relationship among their constituents. | (Nakagawa et al., 2015) |
| **System-of-Systems Type** | Categorization of SoS according to the extent of managerial independence presented by its constituents. | (Lane, 2013) |
| **Centralized System-of-Systems** | This type of SoS is managed by a central authority. | Researchers |
| **Decentralized System-of-Systems** | This type of SoS is not managed by a central authority. | Researchers |
| **Virtual System-of-Systems** | This class of SoS lacks a central management authority and a clear purpose. It is ad hoc and the constituent systems may not be known. | (Lane, 2013) |
| **Collaborative System-of-Systems** | In this class of SoS, teams work together to fulfill commonly agreed purposes. There is no SoSE team to guide or manage the SoS-related activities of the constituent systems. | (Lane, 2013) |
| Acknowledged System-of-Systems | This class of SoS has recognized objectives, a manager, and resources at SoS level that does not have complete authority over the constituent systems. | (Lane, 2013) |
| Directed System-of-Systems | This class of SoS is centrally managed and built to fulfill specific purposes. The constituent systems maintain their ability to operate independently, but the evolution is controlled by the SoS management. | (Lane, 2013) |
| Stakeholder | Individual or organization having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations | ISO/IEC/IEEE 24765 |
| Constituent Stakeholder | Individual or organization having a right, share, claim, or interest in a constituent system or in its possession of characteristics that meet their needs and expectations | ISO/IEC/IEEE 24765 |
| System-of-Systems Stakeholder | Individual or organization having a right, share, claim, or interest in a system-of-systems or in its possession of characteristics that meet their needs and expectations. | ISO/IEC/IEEE 24765 |
| System-of-Systems Engineering | Activities related to SoS design, implementation, and maintenance. | (DoD, 2008) |
| Software Architecture | Structure of software elements, their relationship, and the principles and guidelines governing their evolution over time. | ISO/IEC/IEEE 42010 |
| Environment | The context determining the setting and circumstances of all influences upon a system. | ISO/IEC/IEEE 42010 |
| Quality Attribute | Feature or characteristic that affects the quality of a SoS. | ISO/IEC 25010 |
| Application Domain | An area of interest or expertise in which the SoS pertains. | ISO/IEC/IEEE 24765 |

After the glossary was built, the next task was to define relationships between the concepts. Firstly, taxonomies were established, and the concepts were classified, defining classes and subclasses. From the glossary, the following taxonomies were identified:
Table 3.2: OntoSoS Taxonomies

| Class                                      | Subclasses                                                                 |
|--------------------------------------------|---------------------------------------------------------------------------|
| Mission                                    | Individual Mission, Global Mission                                         |
| System-of-Systems Characteristic           | Complexity, Evolutionary Development, Size, Variability, Emergent Behavior |
| Constituent Characteristic                 | Operational Independence, Managerial Independence, Physical Distribution, Heterogeneity, Reuse |
| System-of-Systems Type                     | Centralized System-of-Systems, Decentralized System-of-Systems             |
| Centralized System-of-Systems              | Acknowledged System-of-Systems, Directed System-of-Systems                 |
| Decentralized System-of-Systems            | Virtual System-of-Systems, Collaborative System-of-Systems                 |
| Stakeholder                                | Constituent Stakeholder, System-of-systems Stakeholder                    |
| Software Architecture                      | Dynamic Architecture                                                       |

With the taxonomies identified, the next step was to build binary relationships between concepts. Here, two concepts (source concept and target concept) were related and a verb was used to establish such relationship. These relations were mostly suggested by researchers in the SoS field and also refined in the OntoSoS evaluation. For each relation, an inverse relation was also defined. In Table 3.3, the relations of the OntoSoS are presented.
### Table 3.3: OntoSoS Relations

| Source Concept                  | Relation       | Target Concept          | Inverse                      |
|---------------------------------|----------------|-------------------------|------------------------------|
| System-of-Systems               | is applied to  | Application Domain      | applies to                   |
| System-of-Systems               | accomplishes  | Global Mission          | is accomplished by           |
| System-of-Systems               | has            | System-of-Systems Stakeholder | is stakeholder of         |
| System-of-Systems               | is classified as | System-of-Systems Type | classifies                   |
| System-of-Systems               | is situated in | Environment             | situates                     |
| System-of-Systems               | presents       | System-of-Systems Characteristic | is presented by        |
| System-of-Systems               | exhibits       | Software Architecture   | is exhibited by              |
| Constituent                     | participates in | System-of-Systems       | has                         |
| Constituent                     | accomplishes  | Individual Mission      | is accomplished by           |
| Constituent                     | has            | Constituent Stakeholder | is stakeholder of            |
| Constituent                     | presents       | Constituent Characteristic | is presented by   |
| System-of-Systems Stakeholder   | is interested in | Global Mission      | is interest of               |
| Constituent Stakeholder         | is interested in | Individual Mission | is interest of               |
| Global Mission                   | satisfies      | Quality Attribute       | is satisfied by              |
| Software Architecture           | encompasses    | Quality Attribute       | is encompassed by            |
| System-of-Systems Engineering   | provides       | System-of-Systems       | is provided by               |
| Managerial Independence         | defines        | System-of-Systems Type  | is defined by                |
| Emergent Behavior               | satisfies      | Global Mission          | is satisfied by              |
| Centralized                     | is managed by  | Central Authority       | manages                      |
| System-of-Systems               | leads to       | Dynamic Architecture    | is led by                    |
| Evolutionary Development        | leads to       | Dynamic Architecture    | is led by                    |

After defining the relations between concepts, an UML class diagram was built for OntoSoS. The diagram also helped researchers when evaluating the ontology concepts and relations. In Figure 3.1, the UML diagram of OntoSoS is presented. Here, only the main relations are presented, the inverse relations are not shown, aiming a clear presentation.
Figure 3.1: OntoSoS - UML Class Diagram
CHAPTER 3. ONTOSOS: AN ONTOLOGY FOR SYSTEMS-OF-SYSTEMS

Formalization

This activity turns a conceptual model into a formal model. When using an ontology editor, this activity is not mandatory (Gómez-Pérez et al., 2004). Since Protégé was used to implement OntoSoS, then this activity was skipped.

Implementation

The language adopted for OntoSoS is OWL, since it is a W3C recommendation and a Semantic Web standard (W3C, 2012a). Moreover, it is the most used language for implementing ontologies. Protégé also generates OWL code and aims the creation and edition of ontologies in a complete environment, providing an user interface. Visualization tools, such as OWL Viz, allow navigation in the classes and relationships of the ontology. The terms were inserted in Protégé as classes, subclasses, and properties. OntoSoS was graphically built and the OWL code was generated automatically.

Firstly, all classes and their subclasses were created in the tool, respecting the taxonomy defined. The Protégé OWL Tutorial recommends that all class names should start with a capital letter and should not contain spaces, so the OntoSoS classes followed this recommendation. In Figure 3.2, the list of classes is shown, implemented in Protégé. The class “Thing” is the class that represents the set containing all individuals. Thus, all classes are subclasses of “Thing”. When creating the classes in Protégé, they were also set as disjoint. This ensures that an individual which has been asserted to be a member of one of the classes cannot be a member of any other class.

Besides the class and subclass relationships, classes were related as object properties in Protégé, according to the relationships defined in Table 3.3. The name of the properties also had to be adapted to meet the recommendations for naming properties. It is recommended that property names start with a lower case letter, have no spaces, and have the remaining words capitalized.

In Figure 3.3, the list of object properties is shown, implemented in Protégé, including the inverse properties. The OWL properties appear sorted in alphabetical order. Since many verbs appear more than once, it is recommended to include in the property name the target class it is related to. In Figure 3.4, a visualization of OntoSoS is shown, using the Protégé plug-in OWL Viz. The figure allows the visualization of the taxonomies created, including the class “Thing”. The subclasses are related to their parent classes by a parent-child relationship, with the label “is-a”.

2http://protegewiki.stanford.edu/wiki/OWL Viz
3http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf
Maintenance

This activity updates and corrects the ontology, if needed. Similarly to any system that is released, it is an ongoing activity that will be carried out from now on.

Support Activities

The following tasks represent the support activities from METHONTOLOGY, discussed as follows:

Acquisition

The knowledge represented in the ontology was acquired from publications (including papers, books, and technical reports) and within the researchers in the SoS field, including the research group START.
Since there is no ontology with similar purpose and degree of formality of OntoSoS, this activity has not been conducted.

**Evaluation**

During the ontology development, an ongoing evaluation was performed, as suggested by METHONTOMETRY. In the ontology modeling phase, regular meetings with researchers were scheduled, so the capture of terms, relationships, and definitions were evaluated. A semi-formal model, an UML diagram, was created to facilitate this process. In order to
CHAPTER 3. ONTOSOS: AN ONTOLOGY FOR SYSTEMS-OF-SYSTEMS

support the consistency checking during the ontology implementation activity, the Protégé plugin Pellet Reasoner \(^4\) was used.

\(^4\)http://clarkparsia.com/pellet/protege/

Figure 3.4: Protégé - OntoSoS Visualization with OWLViz
CHAPTER 3. ONTOSOS: AN ONTOLOGY FOR SYSTEMS-OF-SYSTEMS

Documentation

As required by METHONTOLOGY, all outputs of each activity were properly documented.

Configuration Management

All documents and code generated are versioned using Apache Subversion\(^5\) (SVN), and the repository is hosted at a server at ICMC/USP\(^6\).

3.3 Final Remarks

In this chapter the modelling and implementation of OntoSoS were presented, including all steps followed to reach to the final version of the ontology. The next chapter presents the process conducted to evaluate OntoSoS, which was performed through a survey with researchers.

\(^5\)https://subversion.apache.org/
\(^6\)http://www.labes.icmc.usp.br/svn/ontosos/
OntoSoS Evaluation

4.1 Initial Remarks

The evaluation of OntoSoS was conducted through a survey, and artifacts related to the ontology were submitted to researchers in the SoS field. To create the survey, Google Forms\(^1\) was adopted. The questions were created, and each question had a text area for the participants to give their opinion. In this chapter, the survey structure is presented, explaining how the questions were designed. Then, results of the survey are presented and discussed, and an analysis of the suggestions provided by the researchers is performed.

4.2 Survey Structure

The evaluation process was done based on artifacts produced during the ontology conceptualization. The participants, ten academic researchers, received by e-mail the list of concepts (glossary) and their definitions, as well as the UML Class diagram containing the relationships between the concepts. In Table 4.1 the glossary sent to the researchers is presented. The diagram sent, containing the concepts and relationships is presented in Figure 4.1.

\(^1\)https://docs.google.com/forms/
Figure 4.1: OntoSoS - UML Class Diagram for Evaluation
Together with these artifacts, they received the survey URL, which led them to the survey form.

**Table 4.1:** OntoSoS Glossary for Evaluation.

| Name                          | Description                                                                 |
|-------------------------------|-----------------------------------------------------------------------------|
| System-of-Systems             | A large system that delivers unique capabilities, formed by the interoperation of independently useful systems. |
| Constituent                   | Each independent system that composes the SoS.                              |
| Central Authority             | Entity coordinating the collaboration among constituents towards accomplishing a given goal. |
| Mediator                      | Architectural element that performs the required data translations and coordinate constituents’ behavior. |
| Mission                       | The goal to be accomplished by a SoS.                                       |
| Individual Mission            | Mission assigned to the constituent systems.                                |
| Global Mission                | Mission assigned to the SoS.                                                |
| System-of-System Characteristic| Key aspect related to an SoS.                                               |
| System-Level Characteristic   | Characteristic that describe the system (SoS) as a whole.                  |
| Subsystem-Level Characteristic| Characteristics that primarily describe the system (SoS) in terms of the characteristics of its subsystems (constituents). |
| System Complexity             | The degree to which a system is difficult for its stakeholders to understand and analyze, especially due to having a large number of components connected by many complicated interfaces. |
| System Evolution              | The degree to which (in terms of rate and impact) the goals and requirements for a system (and its subsystems) change over time. |
| System Negative Emergence     | The degree to which the new behaviors and characteristics of a system that result (i.e., emerge) from the interaction of the system’s subsystems are detrimental, unintended, and difficult to predict from the behaviors and characteristics of these individual subsystems. |
| System Size                   | The amount or magnitude of the system with regard to a suitable dimension.  |
| **System Variability** | The degree to which a single type of system simultaneously exists in multiple variants, versions, or configurations. |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| **Subsystem Autonomy** | The degree to which the subsystems within a system are independent, stand alone and are individually useful, self-contained, and operationally independent (i.e., neither controlled by nor controlling other subsystems). |
| **Subsystem Governance** | The degree to which the subsystems of a system are governed (e.g., specified, managed, funded, developed, owned, operated, maintained, and sustained) in an independent, decentralized, and uncoordinated manner. |
| **Subsystem Heterogeneity** | The degree to which the subsystems of a system differ from each other in that they (1) have different goals, objectives, and requirements, (2) have different behavior and characteristics, (3) provide unrelated functionality, (4) belong to different application domains, and (5) are implemented using different technologies. |
| **Subsystem Physical Distribution** | The degree to which the subsystems of a system exist in different physical locations. |
| **Subsystem Reuse** | The degree to which the subsystems of the system have been reused regardless as to whether they are commercial-off-the-shelf (COTS), government-off-the-shelf (GOTS), military-off-the-shelf (MOTS), organizational-internal reuse, open source, and freeware. |
| **System-of-Systems Type** | Categorization of SoS according to the extent of managerial independence presented by its constituents. |
| **Centralized System-of-Systems** | This type of SoS is managed by a central authority. |
| **Decentralized System-of-Systems** | This type of SoS is not managed by a central authority. |
| **Virtual** | This class of SoS lacks a central management authority and a clear purpose. It is ad hoc and the constituent systems may not be known. |
| Collaborative | In this class of SoS, teams work together to fulfill commonly agreed purposes. There is no SoSE team to guide or manage the SoS-related activities of the constituent systems. |
| Acknowledged | This class of SoS has recognized objectives, a manager, and resources at SoS level that does not have complete authority over the constituent systems. |
| Directed | This class of SoS is centrally managed and built to fulfill specific purposes. The constituent systems maintain their ability to operate independently, but the evolution is controlled by the SoS management. |
| Stakeholder | Individual or organization having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations. |
| Stakeholder of Constituent | Individual or organization having a right, share, claim, or interest in a constituent system or in its possession of characteristics that meet their needs and expectations. |
| Stakeholder of SoS | Individual or organization having a right, share, claim, or interest in a system-of-systems or in its possession of characteristics that meet their needs and expectations. |
| System-of-Systems Engineering | Activities related to SoS design, implementation, and maintenance. |
| System Architecture | The conceptual model of a system together with models derived from it that represent (1) different viewpoints defined as views on top of the conceptual model, (2) facets or concerns of the system in dependence on the scope and abstraction level of various stakeholders, (3) restrictions for the deployment of the system and description of the quality warranties of the system, and (4) embeddings into other (software) systems. |
Software Architecture: Structure of software elements, their relationship, and the principles and guidelines governing their evolution over time.

Dynamic Architecture: It refers to architectures that change at runtime; i.e., their overall structure can be modified by adding, replacing, or withdrawing constituents, and even by changing relationship among their constituents.

Environment: The context of hardware, software, and stakeholders influencing the SoS.

Quality Attribute: Feature or characteristic that affects the quality of a SoS.

Application Domain: An area of interest or expertise in which the SoS pertains.

Military: The classification for domains related to defense systems.

Civil: The classification for domains which are non-military.

The OntoSoS evaluation survey was divided into five sections.

- **Introduction**: The goals of the evaluation were presented to the participants, and an option to receive the evaluation results by e-mail was given.

- **Description of Concepts**: The participants evaluated the description of the terms present in the glossary. If the participants strongly disagreed or disagreed with any description, they could indicate which concepts should have the description changed. For that, a list of all concepts was presented with check boxes to select the ones that had to be reviewed. At the end of the section, a space for additional comments was given to the participant.

- **Ontology Coverage**: The participants evaluated if the concepts in ontology correctly represented the Systems-of-Systems field. If the participants strongly disagreed or disagreed with any concept, they could indicate which concepts should have been corrected in the ontology. For that, a list of all concepts was presented with check boxes to select the ones that had to be reviewed. At the end of the section, a space for additional comments was given to the participant.

- **Relationships**: The participants evaluated if the relationships between concepts were correctly mapped. If the participants strongly disagreed or disagreed with any concept, they could indicate which relationships should have been corrected in the
ontology. For that, a list of the relationships was presented with check boxes to select the ones that had to be reviewed. At the end of the section, a space for additional comments was given to the participant.

- **Additional Information:** The participants could give their option about the ontology and explain how he or she was planning to use the ontology. Finally, a text area was provided so the participant could give any additional comment regarding the ontology or the evaluation itself.

A summary of survey sections, questions, and possible answers is presented in Table 4.2. The complete survey structure can be found in Appendix B.

**Table 4.2: Survey Questions**

| Section                  | Question / Sentence                                                                                                                                                                                                 | Possible Answers                                                                 | Comments                                                                 |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|------------------------------------------------------------------------|
| Introduction             | “This evaluation survey is aimed at expert researchers in Systems-of-Systems. The purpose of this survey is to evaluate the concepts represented and how they are related in OntoSoS, an ontology for systems-of-systems. The survey contains 14 questions and your participation is anonymous. If you want to be informed about the final result, please enter a valid e-mail address below.”. | Enter E-mail                                                                   | N/A                                                                    |
| Description of Concepts   | “The ontology correctly describes all concepts related to Systems-of-Systems.”.                                                                                                                                  | Strongly Disagree, Disagree, Undecided, Agree, Strongly Agree                  | “If you strongly disagree or disagree with any description, please explain why.” |
| Ontology Coverage        | “All relevant concepts related to the Systems-of-Systems field have been represented in the ontology.”.                                                                                                          | Strongly Disagree, Disagree, Undecided, Agree, Strongly Agree                  | “If you strongly disagree or disagree, please explain which concepts should be corrected or included in the ontology.” |

46
### Relationships

“*All relevant relationships related to the Systems-of-Systems concepts have been correctly mapped in the ontology.*”

| Strongly Disagree, Disagree, Undecided, Agree, Strongly Agree |
| "If you strongly disagree or disagree, please explain which relationships should be corrected or included in the ontology." |

| Yes, No |
| If not, please elaborate your answer. |

| Yes, No |
| If yes, please explain how do you plan to use it. |

| Free text |
| N/A |

### 4.3 Results

After receiving the ontology artifacts and the survey, the participants could analyze the content provided to answer the survey and give their opinion. In the survey, ten researchers in the SoS field were selected to answer questions regarding the ontology.

The results of the evaluation are presented and discussed according to the survey sections.

#### Results: Description of Concepts

In Figure 4.2, a summary of the evaluation regarding the description of concepts is presented. Half of researchers’ said that the ontology described correctly all concepts of SoS and 40% of the researchers declared to be undecided about the descriptions. Only 10% disagreed on the descriptions. Due to the results, some concepts had their descriptions enhanced or changed, adopting a different description (contemplating a different author or reference for the given concept).
CHAPTER 4. ONTOSOS EVALUATION

Results: Ontology Coverage

In Figure 4.3, a summary of the evaluation regarding the ontology coverage is presented. More than half of researchers’ opinions (60%) said that all relevant concepts related to SoS field were presented in the ontology, 20% of the researchers declared undecided, and 20% disagreed of the coverage proposed. Due to the results and suggestions, the ontology coverage was reviewed, and some concepts were removed, while others have been added.

Results: Ontology Relationships

In Figure 4.4, a summary of the evaluation regarding the ontology relationships is presented. More than half of researchers’ opinions (60%) said that all relevant relationships related to SoS concepts were correctly mapped in the ontology, while 40% of the researchers disagreed of the relationships proposed. In this evaluation section, no researchers declared undecided. Due to the results and suggestions, some ontology relationships were reviewed, having the verb changed. Other relationships have been created or removed, based on the suggestions provided.
Results: Additional Information

In this evaluation section, the participants could evaluate the need of an ontology for Systems-of-Systems and also if they would consider using the ontology in future research. In Figures 4.5 and 4.6, a summary of the answers is presented. All of the participants declared that an ontology for Systems-of-Systems is required, which demonstrates that this field needs clarification and establishment of the concepts. Apart from that, 90% of the researchers would consider using OntoSoS in their next projects, which demonstrates they can see the value that such ontology brings to the research field.

The answers received were grouped using the concepts of Coding (Corbin and Strauss, 2008), in which the suggestions could be separated into groups, according to researchers’ opinions. Then, for each suggestion, a code was created (Corbin and Strauss, 2008). It facilitated the process of grouping and applying the suggestions, in the case different researchers proposed the same change. The groups and subgroups were created based on the attributes of the evaluation:
Figure 4.6: Researchers’ Will in Using OntoSoS on Their Next Projects

- **Concept Name:** This group indicates that an researchers suggested something related to a concept name of the glossary. The only subgroup identified here was Change Concept Name. For this group, the code CN was established.

- **Description of Concept:** This group indicates that an researcher suggested something related to the description of the concept in the glossary. The only subgroup identified here was Change Description of Concept. For this category, the code DC was established.

- **Relationship:** This group indicates that an researcher suggested something related to the relationships between concepts in the diagram. For this group, the following subgroups were identified: Change Taxonomy, Change Relationship Name, Establish New Relationship, Remove Relationship, and Change Relationship. For this group, the code R was established.

- **Concept / Relationship:** This group indicates that an researcher suggested something related to both a concept and a relationship. The only subgroup identified here was Establish Different Concepts and Taxonomy. In this case, the researcher suggested to have a taxonomy with different concepts. For this group, the code CR was established.

- **Ontology Coverage:** This group indicates that an researcher suggested something related to the coverage of the ontology. For this group, the following subgroups were identified: Propose New Concepts, Remove Concept, Remove Concept Property, Insert Concept, and Change Concept. For this group, the code OC was established.

A summary of the groups, subgroups, and codes created is presented in Table 4.3.

Table 4.3: Evaluation Groups, Subgroups, and Codes
For each code, a suffix with an index indicates the count for it (for instance, CN1, CN2, and so on). All groups, subgroups, codes, suggestions, and the indication if it was accepted are presented in Table 4.4. A duplicated code indicates that more than one researcher provided the same suggestion. From the suggestions provided, 39 were accepted, while 26 were not accepted.

**Table 4.4: Researchers’ Suggestions**

| Group               | Subgroup                  | Code | Suggestion                                                                 | Accept? |
|---------------------|---------------------------|------|-----------------------------------------------------------------------------|---------|
| Concept Name        | Change Concept Name       | CN1  | Replace *System Negative Emergence* by Emerging Behavior                    | Yes     |
| Concept Name        | Change Concept Name       | CN1  | Replace *System Negative Emergence* by Emerging Behavior                    | Yes     |
| Concept Name        | Change Concept Name       | CN2  | Replace *System-of-Systems* by System                                        | No      |
| Concept Name        | Change Concept Name       | CN3  | Change *Subsystem*                                                          | Yes     |
| Concept Name        | Change Concept Name       | CN3  | Change *Subsystem*                                                          | Yes     |
| Concept Name        | Change Concept Name       | CN3  | Change *Subsystem* (replace by Constituent System)                          | Yes     |
| Concept Name        | Change Concept Name       | CN3  | Change *Subsystem*                                                          | Yes     |
| Concept Name        | Change Concept Name       | CN4  | Change *Type*                                                               | No      |
| Concept Name        | Change Concept Name       | CN5  | Replace *Autonomy* by Autonomy                                              | Yes     |
| Description of Concept | Change Description of Concept | DC1 | Change *Subsystem Physical Distribution* description. | No |
|------------------------|--------------------------------|-----|------------------------------------------------------|----|
| Description of Concept | Change Description of Concept | DC1 | Change *Subsystem Physical Distribution* description | No |
| Description of Concept | Change Description of Concept | DC2 | Change *System-of-Systems* description | Yes |
| Description of Concept | Change Description of Concept | DC3 | Change *Environment* description | Yes |
| Description of Concept | Change Description of Concept | DC3 | Change *Environment* description | Yes |
| Description of Concept | Change Description of Concept | DC3 | Change *Environment* description | Yes |
| Description of Concept | Change Description of Concept | DC3 | Change *Environment* description | Yes |
| Description of Concept | Change Description of Concept | DC4 | Change *System-of-Systems Engineering* description | Yes |
| Description of Concept | Change Description of Concept | DC4 | Change *System-of-Systems Engineering* description | Yes |
| Description of Concept | Change Description of Concept | DC5 | Change *Mission* description | Yes |
| Description of Concept | Change Description of Concept | DC5 | Change *Mission* description | Yes |
| Description of Concept | Change Description of Concept | DC6 | Change *Global Mission* description | Yes |
| Description of Concept | Change Description of Concept | DC6 | Change *Global Mission* description | Yes |
| Description of Concept | Change Description of Concept | DC7 | Change *Directed* description | No |
| Description of Concept | Change Description of Concept | DC8 | Change *Acknowledged* description | No |
| Description of Concept | Change Description of Concept | DC9 | Change *Virtual* description | No |
| Description of Concept | Change Description of Concept | DC10 | Change *Collaborative* description | No |
| Description of Concept | Change Description of Concept | DC11 | Change *Subsystem Autonomy* description | No |
| Description of Concept | Change Description of Concept | DC | Change System Variability description | Yes/No |
|------------------------|-------------------------------|----|--------------------------------------|--------|
| Description of Concept | Change Description of Concept | DC12 | Change Stakeholder of Constituent description | No |
| Description of Concept | Change Description of Concept | DC13 | Change Stakeholder of SoS description | No |
| Description of Concept | Change Description of Concept | DC14 | Change Individual Mission description | Yes |
| Description of Concept | Change Description of Concept | DC15 | Change Subsystem-Level Characteristic description | No |
| Description of Concept | Change Description of Concept | DC16 | Change Military description | Yes |
| Relationship Change Taxonomy | R1 | Put Subsystem Physical Distribution as subclass of System-Level Characteristic | No |
| Relationship Change Taxonomy | R1 | Put Subsystem Physical Distribution as subclass of System-Level Characteristic | No |
| Relationship Change Relationship Name | R2 | Replace System-of-Systems realizes Mediator by System-of-Systems is supported by Mediator | Yes |
| Relationship Change Relationship Name | R3 | Replace System-of-Systems realizes Mediator by System-of-Systems uses Mediator | Yes |
| Relationship Change Relationship Name | R4 | Replace System-of-Systems realizes Mediator by System-of-Systems communicate through Mediator | Yes |
| Relationship Change Relationship Name | R5 | Replace System Architecture enables Quality Attribute by System Architecture encompasses Quality Attribute | Yes |
| Relationship Change Relationship | R8 | Change relationships between System architecture, Software Architecture, and Dynamic Architecture | Yes |
| Relationship Change Relationship Name | R13 | Change relationship name System-of-Systems Engineering develops System-of-Systems | Yes |
| Relationship Change Relationship Name | R13 | Change relationship name System-of-Systems Engineering develops System-of-Systems | Yes |
| Relationship Establish New Relationship | R6 | Insert relationship System Variability leads to Dynamic Architecture | Yes |
| Relationship | Establish New Relationship | R7 | Establish relationship between Constituent and Subsystem-Level-Characteristic | Yes |
| Relationship | Establish New Relationship | R7 | Establish relationship between Constituent and Subsystem-Level-Characteristic | Yes |
| Relationship | Establish New Relationship | R12 | Insert relationship between Mission and Quality Attribute | Yes |
| Relationship | Establish New Relationship | R17 | Insert relationship between Environment and Stakeholder | No |
| Relationship | Establish New Relationship | R18 | Insert relationship between System-of-Systems and Central Authority | No |
| Relationship | Establish New Relationship | R19 | Insert relationship between Stakeholder of Constituent and Global Mission | No |
| Relationship | Establish New Relationship | R20 | Insert relationship between Stakeholder of SoS and Individual Mission | No |
| Relationship | Establish New Relationship | R14 | Insert relationship between System Architecture and System Variability | No |
| Relationship | Establish New Relationship | R15 | Insert relationship between Environment and Constituent | No |
| Relationship | Establish New Relationship | R9 | Insert relationship between Emergent Behavior and Global Mission | Yes |
| Relationship | Remove Relationship | R11 | Remove relationship between Stakeholder and Mission | Yes |
| Concept / Relationship | Establish Different Concepts and Taxonomy | CR1 | Propose a new classification for types of Systems-of-Systems. | No |
| Ontology Coverage | Propose New Concepts | OC1 | Propose mechanisms to relate Constituent and Mediator | No |
| Ontology Coverage | Remove Concept | OC2 | Remove concept Civil | Yes |
| Ontology Coverage | Remove Concept | OC3 | Remove concept Militar | Yes |
| Ontology Coverage | Remove Concept | OC4 | Remove property degreeOfControl | Yes |
| Ontology Coverage | Remove Concept | OC5 | Remove concept Subsystem Reuse | No |
The changes performed were important to refine OntoSoS and meet more precisely the needs of SoS community in terms of lack of consensus and terminology. All changes performed resulted in OntoSoS, presented in Chapter 3.

The justification for the non-accepted suggestions, organized by codes, are summarized in Table 4.5.

**Table 4.5: Justification for Suggestions Not Accepted**

| Codes         | Justification                                                                                                                                 |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| DC1, DC11, DC12, DC16 | These definitions were taken from the study of Firesmith (2010) and are already widespread in the community.                                      |
| R1           | As defined by Firesmith (2010), the subsystem physical distribution refers to the constituents of the SoS.                                        |
| DC7, DC8, DC9, DC10, CR1 | The classification and descriptions provided by Lane (2013) are already well accepted by the community.                                         |
| CN2          | The term system can also refer to a SoS. However, the main purpose of the ontology is to establish SoS concepts, so it must be explicit here.       |
| OC1          | There is technology specification in the ontology.                                                                                            |
| DC17, R14    | Concepts have been removed from the ontology.                                                                                                 |
| OC5, OC6     | They are common characteristics to Systems-of-Systems and they can be used to determine if a system is a SoS.                                  |
| OC7          | The concept is already covered by emergent behavior, which is present in the ontology.                                                         |
CHAPTER 4. ONTOSOS EVALUATION

| OC9  | The concept is a general thing, not considered as a concept to SoS. |
| DC13, DC14 | The definitions are similar to common systems, and they were taken from ISO/IEC/IEEE 24765. |
| CN14 | “Type” is already used to classify the SoS. |
| R15 | The constituent in part of the SoS, which is already related to the environment itself. |
| R17 | The environment proposed is where the SoS is present, it is not related to the stakeholders. |
| R18 | There is a specific type of SoS (Centralized) that is related to the Central Authority. |
| R19, 20 | We want to differentiate the stakeholders, so each type is interested in a type of mission. |

Furthermore, the some ideas for using OntoSoS in future projects were provided by the researchers in the survey:

- Establishment of a common vocabulary and presentation of the different dimensions and concepts of SoS. Since SoS is a research field very new and several concepts are not still broadly defined, an ontology is very important to establish standardized concepts about this research area to the whole community.

- Support to the requirements engineering for SoS. This activity could start by the identification of the relevant elements depicted in the ontology. Hence, an ontology-driven requirements engineering could take place, and the software architecture development could also exploit the concepts presented as a representation of the main parts of an SoS.

- Common understanding and communication among multiple stakeholders and different development processes. Since the concepts are presented and defined, it could help to clarify gaps in understanding and communication.

### 4.4 Final Remarks

In this chapter the first version of OntoSoS was presented. After submitting the ontology to a process of evaluation, many suggestions for the ontology improvement and refinement were provided. In order to organize and facilitate this process, the suggestions were divided into groups, subgroups, and codes. The researchers’ suggestions were collected through a survey, separated into sections with different goals. The ontology was evaluated according
to the concept names and descriptions, the relationships, and the coverage. At the end of
the survey, researchers could provide additional information, also indicating the will to use
the ontology in future projects. After analyzing the researchers’ opinions, the ontology
was reviewed and enhanced, resulting in a new version, which is presented in Chapter 3.

The next chapter focuses on presenting a conclusion of this Master project, indicating
future work and perspectives to use OntoSoS.
5.1 Characterization and Contributions of the Research

As an emerging research field, the demand for SoS is growing. There are many definitions, concepts, and terms in this field that are not well-established, leading to a lack of common understating among researchers and practitioners. As stated by Jamshidi (2008a), there is a great variability in the literature, definitions and perspectives, which makes it difficult in advancing and understanding the field.

In parallel, ontologies can be used to represent the knowledge related to a given area by making such knowledge explicit and shared. An ontology for SoS could help in establishing a common understanding of the field, facilitating the knowledge sharing and contributing to the evolution and consolidation of such field.

In this Master’s project, OntoSoS, an ontology for SoS, was established. This ontology describes terms and concepts related to the SoS field. In this perspective, a common vocabulary was defined, so a common understanding of the information can be shared among practitioners and researchers, bringing the reuse of knowledge and making definitions explicit. The main contributions of this Master’s project are the following ones:

1. Consolidation of the SoS field, by establishing a consensus about the knowledge, facilitating communication and knowledge sharing;

2. Contribution to the SoS field and to SoSE with the establishment of a vocabulary that will make it possible to improve the understanding;
3. Help to guide software engineers when building Systems-of-Systems; and

4. Contribution to SoS learning by providing a common understanding for terms and concepts used in SoS community.

5.2 Difficulties and Limitations

The most exhausting task when creating OntoSoS was the phase of conceptualization. The main reason for it is that the first list of the concepts was conceived, and many meetings with the research group were conducted. Each researcher had a different point of view, and during the meetings, they were willing to concentrate and guide the discussions to their own research subjects.

Another challenge when building the ontology was faced during the evaluation. Even setting a distant deadline for answering the survey, some of the researchers took too much time to respond and, consequently, the analysis also took more time. There were cases where the answers for some researchers would not be considered because of no responses, and after the deadline passed, the researchers responded the survey. Then the analysis had to be repeated to include the new suggestions.

Limitations can also be identified in the artifacts sent to researchers for evaluation. The ontology class diagram did not have the cardinality between the relationships, so it might have affected the interpretation in some of those relationships.

Regarding the evaluation results, some limitations can be observed. The suggestions given by the researchers had to be summarized and grouped. During the analysis of the answers, some suggestions might be lost or mixed, and then the results of the evaluation could be affected. Furthermore, some suggestions contradicted well-accepted studies in the field, then those suggestions were not accepted. Along with that, the answers had to be mapped to feasible suggestions in the ontology. This mapping process might be not accurate for some cases, affecting the results of the evaluation as a whole.

5.3 Future Work

During the development of this Master’s project, different opportunities could be identified to continue this research initiated.

The next steps for OntoSoS are for sure to keep improving the ontology, by adding more concepts, relationships, and increasing the ontology coverage. One of the possible approaches for that is to propose an evaluation in the industry, with software engineers and practitioners. By conducting such process, the suggestions from the researchers in the
area could be compared to the ones in the industry. Hence, OntoSoS could be enhanced in different directions.

An interesting approach is to use OntoSoS as a learning material in Software Engineering and Software Architecture courses. Since students usually are not familiar with SoS concepts, an experiment could be conducted to evaluate the knowledge obtained by using the ontology. For instance, a questionnaire could be used to evaluate the knowledge after exploring the ontology.

Once OntoSoS gets more visibility, it could be used to help software engineers in building new SoS, by guiding them from requirements analysis to maintenance of SoS and its constituents. This approach can also be extended to Cyber-Physical Systems (Sanislav and Miclea, 2012).

Since OntoSoS is implemented in OWL, it can be integrated and accessed by machines. A web interface could be created to access the ontology and explore the concepts. It could also enable the collaboration among researchers in enhancing the ontology and proposing improvements. Along with that, the system could have a search engine so the end users could use the ontology to learn the concepts and search for relations between the concepts. For instance, once an user searches for a given term, it would return the definition and possible relations with this term.

Finally, OntoSoS could be also used as a SoS repository. Once integrated with a system, software engineers could instantiate new SoS using the ontology. Then, every time a new SoS is built among a group, it would be instantiated in the ontology, including also instances for its constituents, stakeholders and so on. Then the number of instances in the ontology would increase as new SoS are conceived.
Abdalla, G.; Damasceno, C. D. N.; Guessi, M.; Oquendo, F.; Nakagawa, E. Y. A systematic literature review on knowledge representation approaches for systems-of-systems. In: 2015 IX Brazilian Symposium on Components, Architectures and Reuse Software, 2015a, p. 70–79.

Abdalla, G.; Damasceno, C. D. N.; Nakagawa, E. Y. A Systematic Literature Review on Systems-of-Systems Knowledge Representation. Technical Report 405, Department of Computer Science, University of São Paulo, São Carlos, São Paulo, Brazil, 2015b.

Autran, F.; Auzelle, J.-P.; Cattan, D.; Garnier, J.-L.; Luzeaux, D.; Mayer, F.; Peyrichon, M.; Ruault, J.-R. Coupling component systems towards systems of systems. In: 18th Annual International Symposium of INCOSE, The Netherlands, 2008.

Babu, L.; Seetha Ramaiah, M.; Prabhakar, T. V.; Rambabu, D. Archvoc–towards an antology for software architecture. In: Second Workshop on SHAring and Reusing Architectural Knowledge Architecture, Rationale, and Design Intent (SHARK-ADI 2007), Washington, DC, USA: IEEE Computer Society, 2007, p. 5.

Barbosa, E. F.; Nakagawa, E. Y.; Maldonado, J. C. Towards the establishment of an ontology of software testing. In: 18th International Conference on Software Engineering and Knowledge Engineering (SEKE 2006), San Francisco, CA, USA, 2006, p. 522–525.

Barot, V.; Henshaw, M.; Siemieniuch, C.; Dogan, H. Design of a web-based thesaurus for systems of systems engineering. In: 8th International Conference on System of Systems Engineering, Wailea-Makena , HI, USA, 2013, p. 7–12.

Breitman, K. Web semântica: a internet do futuro. 1st. ed. LTC, 2005.
REFERENCES

Butterfield, M. L.; Pearlman, J. S.; Vickroy, S. C. A system-of-systems engineering geoss: architectural approach. *IEEE Systems Journal*, v. 2, n. 3, p. 321–332, 2008.

Corbin, J. M.; Strauss, A. L. *Basics of qualitative research*. 3 ed. Sage, 2008.

Corcho, O.; Fernández, M.; Gómez-Pérez, A.; López-Cima, A. Building legal ontologies with METHONTOLOGY and WebODE. In: *Law and the Semantic Web (LNAI)*, Heidelberg, DE: Springer, 2005, p. 142–157.

Dahmann, J.; Baldwin, K. Understanding the current state of us defense systems of systems and the implications for systems engineering. In: *2nd Annual IEEE Systems Conference (SYSCON 2008)*, Marriott Chateau Champlain Montreal, QC, Canada, 2008, p. 1–7.

DARPA The DARPA Agent Markup Language Homepage. [http://www.daml.org/](http://www.daml.org/), [Online; Accessed 20-June-2017], 2000.

DoD Office of the deputy under decretary of defense for acquisition and technology, systems and software engineering. Systems engineering guide for systems of systems, version 1.0. Washington, DC: ODUSD(A&T)SSE. 2008.

Fensel, D.; van Harmelen, F.; Horrocks, I.; McGuiness, D. L.; Patel-Schneider, P. F. Oil: an ontology infrastructure for the semantic web. *IEEE Intelligent Systems*, v. 16, p. 38–45, 2001.

Fernández-López, M.; Gómez-Pérez, A.; et al. Overview and analysis of methodologies for building ontologies. *The Knowledge Engineering Review*, v. 17, n. 2, p. 129–156, 2002.

Fernández-López, M.; Gómez-Pérez, A.; Juristo, N. Methontology: from ontological art towards ontological engineering. In: *Symposium on Ontological Engineering of AAAI*, 1997, p. 33–40.

Firesmith, D. *Profiling systems using the defining characteristics of systems of systems (sos)*. Technical Report CMU/SEI-2010-TN-001, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 2010.

Gruber, T. R. A translation approach to portable ontology specifications. *Knowledge Acquisition*, v. 5, n. 2, p. 199–220, 1993.

Guarino, N. Formal ontology and information systems. In: Guarino, N., ed. *1st International Conference on Formal Ontologies in Information Systems (FOIS 1998)*, Trento, Italy: IOS Press, 1998, p. 3–15.
Gómez-Pérez, A.; Corcho, O.; Fernandez-Lopez, M.  *Ontological engineering: with examples from the areas of knowledge management, e-commerce and the semantic web*. 1st. ed. Springer, 2004.

van Harmelen, F.; Patel-Schneider, P. F.; Horrocks, I.  Reference description of the DAML+OIL (March 2001) ontology markup language.  http://www.daml.org/2001/03/reference.html, [Online; Accessed 20-June-2017], 2001.

IEEE-Reliability-Society  Technical committee on ‘systems of systems’ – white paper.  2014.

Jamshidi, M.  System-of-systems engineering-a definition.  *IEEE International Conference on Systems, Man and Cybernetics*, p. 10–12, 2005.

Jamshidi, M.  *Introduction to system of systems* John Wiley Sons, Inc., p. 1–20, 2008a.

Jamshidi, M.  System of systems engineering - new challenges for the 21st century.  *IEEE Aerospace and Electronic Systems Magazine*, v. 23, n. 5, p. 4–19, 2008b.

Kapoor, B.; Sharma, S.  A comparative study ontology building tools for semantic web applications.  2010.

Kitchenham, B.; Charters, S.  *Guidelines for performing systematic literature reviews in software engineering*.  Technical Report, Keele University and Durham University Joint Report, 2007.

Available in  http://www.dur.ac.uk/ebse/resources/Systematic-reviews-5-8.pdf

Kriengle, A. J.  *Behind the wizard’s curtain: an integration environment for a system of systems*.  National Defense University Press, 1999.

Lacy, L. W.  *Owl: representing information using the web ontology language*.  Victoria, BC: Trafford Publishing, 2005.

Lane, J. A.  *What is a system of systems and why should i care?*  Technical Report, University of Southern California, Los Angeles, CA, USA, 2013.

Luke, S.; Hefflin, J.  *Shoe 1.01. proposed specification*.  Technical Report, University of Maryland, Maryland, USA, 2000.

Luzeaux, D.; Ruault, J.-R.  *Systems of systems*.  1st. ed.  London: ISTE, 2010.
REFERENCES

Maier, M. W. Architecting principles for systems-of-systems. *Systems Engineering*, v. 1, n. 4, p. 267–284, 1998.

Nakagawa, E. Y.; Oquendo, F.; Avgeriou, P.; Cuesta, C. E.; Drira, K.; Maldonado, J. C.; Zisman, A. Foreword: Towards reference architectures for systems-of-systems. In: *2015 IEEE/ACM 3rd International Workshop on Software Engineering for Systems-of-Systems*, 2015, p. 1–4.

Oberndorf, P.; Sledge, C. Evolution of a software engineer in a SoS system engineering world. In: *4th Annual IEEE Systems Conference (SYSCON 2010)*, Wan Chai, Hong Kong, 2010, p. 91–96.

van Ruijven, L. Ontology for systems engineering. *Procedia Computer Science*, v. 16, p. 383–392, 2013.

Sage, A. P.; Cuppan, C. D. On the systems engineering and management of systems of systems and federations of systems. *Information Knowledge Systems Management*, v. 2, n. 4, p. 325–345, 2001.

Sanislav, T.; Miclea, L. Cyber-physical systems-concept, challenges and research areas. *Journal of Control Engineering and Applied Informatics*, v. 14, p. 28–33, 2012.

Shapiro, S. C. Knowledge representation. In: *Encyclopedia of Cognitive Science*, John Wiley Sons, Ltd, 2006. Available in http://dx.doi.org/10.1002/0470018860.s00058

Silva, E.; Cavalcante, E.; Batista, T.; Oquendo, F. Bridging missions and architecture in software-intensive systems-of-systems. In: *2016 21st International Conference on Engineering of Complex Computer Systems (ICECCS)*, 2016, p. 201–206.

Silva, L. E. P.; Paiva, D. M. B.; Barbosa, E. F.; Braga, R. T. V.; Cagnin, M. I. Onto-resasset development: An ontology for reusable assets specification and management. In: *26th International Conference on Software Engineering and Knowledge Engineering (SEKE 2014)*, Hyatt Regency, Vancouver, Canada, 2014, p. 459–462.

Souza, E.; Falbo, R.; Vijaykumar, N. Using ontology patterns for building a reference software testing ontology. In: *17th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW)*, 2013, p. 21–30.

Studer, R.; Benjamins, V. R.; Fensel, D. Knowledge engineering: principles and methods. *Data & Knowledge Engineering*, v. 25, n. 1-2, p. 161–197, 1998.

64
REFERENCES

UML Unified Modeling Language. http://www.uml.org/, [Online; Accessed 20-June-2017], 2014.

Uschold, M. Building ontologies: towards a unified methodology. In: *16th Annual Conference of the British Computer Society Specialist Group on Expert Systems*, 1996, p. 16–18.

Uschold, M.; Gruninger, M. Ontologies: principles, methods and applications. *Knowledge Engineering Review*, v. 11, p. 93–136, 1996.

Uschold, M.; King, M.; Moralee, S.; Zorgios, Y. The enterprise ontology. *Knowledge Engineering Review*, v. 13, n. 1, p. 31–89, 1998.

W3C OWL web ontology language use cases and requirements. http://www.w3.org/TR/2004/REC-webont-req-20040210/, [Online; Accessed 20-June-2017], 2004a.

W3C Resource Description Framework (RDF). http://www.w3.org/RDF/, [Online; Accessed 20-June-2017], 2004b.

W3C OWL 2 Web Ontology Language Document Overview (Second Edition). https://www.w3.org/TR/owl2-overview/, [Online; Accessed 20-June-2017], 2012a.

W3C OWL 2 Web Ontology Language Profiles. http://www.w3.org/TR/2012/REC-owl2-profiles-20121211/, [Online; Accessed 20-June-2017], 2012b.

W3C RDF 1.1 Concepts and Abstract Syntax. http://www.w3.org/TR/rdf11-concepts/, [Online; Accessed 20-June-2017], 2014a.

W3C RDF Schema 1.1. http://www.w3.org/TR/rdf-schema/, [Online; Accessed 20-June-2017], 2014b.

W3C Web Ontology Language (OWL). http://www.w3.org/OWL/, [Online; Accessed 20-June-2017], 2014c.

Wong, W.; Liu, W.; Bennamoun, M. Ontology learning from text: a look back and into the future. *ACM Computing Surveys*, v. 44, n. 4, p. 20:1–20:36, 2012.
Ontology Requirement Specification Document

This was the initial specification document for OntoSoS. Some items have changed during the process of OntoSoS creation.

Research Field / Domain / Knowledge Area

System-of-Systems (SoS)

Date

April, 2015

Conceptualized by

Gabriel Abdalla
Milena Guessi Margarido
Valdemar Vicente Graciano Neto
Marcelo Benites Gonçalves
Thiago Bianchi
Elisa Yumi Nakagawa
APPENDIX A. ONTOSOS SPECIFICATION

Implemented by
Gabriel Abdalla

Purpose

The main purpose of this document is to present an ontology covering the main con-
cepts of the software-intensive SoS research field. This ontology establishes a common
understanding about SoS in order to facilitate knowledge sharing among researchers and
practitioners in this field, contributing to the maturation and evolution of such field, and
also serving as a learning resource for students and professionals that intend to investigate
and advance this field. In this scenario, the expected end-users of this ontology will be
researchers, practitioners, students, or anyone interested in the field of SoS.

Level of Formality

Formal

Scope

This ontology covers general concepts, characteristics, classifications, application domains,
quality attributes, and technological platforms that have appeared in the development of
software-intensive SoS.

- **General concepts:** System-of-Systems; Constituent system; Mission / Goal.
- **Characteristics:** Operational independence; Managerial independence; Evolution-
ary development and dynamic architecture; Emergent behavior; Distribution; Soft-
ware intensity.
- **Classification:** Virtual; Collaborative; Acknowledge; Directed.
- **Quality attributes**
- **Application domains**
- **Technology platforms** (e.g., Internet of Things, Cloud Computing, Wireless Sen-
sor Network, Big data, Service Orientation, among others)
APPENDIX A. ONTOSOS SPECIFICATION

Sources of Knowledge

- ABDALLA, G.; DAMASCENO, C. D.; GUSSSI, M.; OQUENDO, F.; Nakagawa, E.Y.; A Systematic Literature Review on Knowledge Representation Approaches for Systems-of-Systems, IX Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS 2015), Brazilian Congress on Software (CBSOft 2015), Belo Horizonte, Brazil, 2015, p. 70-79.

- DoD - Office of the deputy under secretary of defense for acquisition and technology, systems and software engineering. Systems engineering guide for systems of systems, version 1.0. Washington, DC: ODUSD (AT) SSE. 2008.

- FIRESMITH, D. Profiling Systems Using the Defining Characteristics of Systems of Systems (SoS), Technical Report CMU/SEI-2010-TN-001, Software Engineering Institute, Carnegie Mellon University, 2010.

- GOLÇALVES, M.B.; GUESSI, M.; OLIVEIRA, L.B.R.; FELIZARDO, K.R.; OQUENDO, F.; NAKAGAWA, E.Y. Software Architectures for Systems-of-Systems: A Systematic Mapping, Information and Systems Technology, 2015, p. 1-64.

- GONÇALVES, M. B.; CAVALCANTE, E.; BATISTA, T.; OQUENDO, F.; NAKAGAWA, E. Y. Towards a Conceptual Model for Software-Intensive Systems-of-Systems. IEEE International Conference on Systems, Man and Cybernetics (SMC), p. 1605-1610, 2014.

- International Standard ISO/IEC/IEEE 24765 Systems and software engineering - Vocabulary, First Edition, 2010.

- International Council on Systems Engineering (INCOSE). Systems Engineering Handbook: A Guide for System Life Cycle Processes and activities. Version 3.2, 2010.

- JAMSHIDI, M. System of Systems Engineering - Innovations for 21st Century. Wiley, 2009.

- KLEIN, J.; VLIET, H. A systematic review of system-of-systems architecture research. In: Proceedings of the 9th international ACM Sigsoft conference on Quality of software architectures (QoSA ’13). p. 13-22. ACM, New York, NY, USA, 2013.

- LANE, J. A. What is a system of systems and why should I care? Technical Report USC-CSSE-2013-001, University of Southern California, 2013.
• MAIER, M. W. Architecting principles for systems-of-systems. Systems Engineering, v. 1, n. 4, p. 267-284, 1998.

• NAKAGAWA, E.Y.; GONÇALVES, M.B.; GUESSI, M.; OLIVEIRA, L.B.R.; OQUENDO, F. The State of the Art and Future Perspectives in Systems-of-Systems Software Architectures. In Proceedings of the First International Workshop on Software Engineering for Systems-of-Systems (SESoS ’13), pp. 13-18. Montpellier, France, 2013.

• NIELSEN, C.B.; LARSEN, P.G.; FITZGERALD, J.; WOODCOCK, J.; PELESKA, J. Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions. ACM Computing Surveys, Vol. 48 (2), September 2015.

• SILVA, E.; BATISTA, T.; OQUENDO, F.. A mission-oriented approach for designing system-of-systems. In: 10th System of Systems Engineering Conference (SoSE), 2015, p. 346-351, May, 2015.
Survey Sections and Questions

This Appendix contains all survey structure, separated by sections, containing all items present in the survey form.
OntoSoS Evaluation

This evaluation survey is aimed at expert researchers in Systems-of-Systems. The purpose of this survey is to evaluate the concepts represented and how they are related in OntoSoS, an ontology for systems-of-systems. The survey contains 14 questions and your participation is anonymous. If you want to be informed about the final result, please enter a valid e-mail address below.

*Required

1. E-mail address (optional):

Description of Concepts

In this section, please evaluate the description of the terms present in the glossary available for the ontology.

2. The ontology correctly describes all concepts related to Systems-of-Systems. *
   Mark only one oval.
   
   [ ] Strongly disagree
   [ ] Disagree
   [ ] Undecided
   [ ] Agree
   [ ] Strongly agree
3. If you strongly disagree or disagree with any description, please indicate which concepts should have the description changed.

Tick all that apply.

- System-of-Systems
- Constituent
- Mediator
- Stakeholder
- Stakeholder of Constituent
- Stakeholder of SoS
- Mission
- Individual Mission
- Global Mission
- System-of-Systems Engineering
- Application Domain
- Military
- Civil
- System-of-Systems Type
- Decentralized System-of-Systems
- Virtual
- Collaborative
- Centralized System-of-Systems
- Directed
- Acknowledged
- Central Authority
- Environment
- Quality Attribute
- Software Architecture
- System Architecture
- Dynamic Architecture
- System-of-System Characteristic
- System-Level Characteristic
- System Complexity
- System Evolution
- System Negative Emergence
- System Size
- System Variability
- Subsystem-Level Characteristic
- Subsystem Autonomy
- Subsystem Governance
- Subsystem Heterogeneity
- Subsystem Physical Distribution
- Subsystem Reuse
4. If you strongly disagree or disagree with any description, please explain why.

........................................................................................................................................................................
........................................................................................................................................................................
........................................................................................................................................................................
........................................................................................................................................................................

Ontology Coverage
In this section, please evaluate if the concepts in ontology correctly represent the Systems-of-Systems field.

5. All relevant concepts related to the Systems-of-Systems field have been represented in the ontology.*

Mark only one oval.

☐ Strongly disagree
☐ Disagree
☐ Undecided
☐ Agree
☐ Strongly agree
6. If you strongly disagree or disagree, please indicate which concepts should be corrected in the ontology.

Tick all that apply.

- System-of-Systems
- Constituent
- Mediator
- Stakeholder
- Stakeholder of Constituent
- Stakeholder of SoS
- Mission
- Individual Mission
- Global Mission
- System-of-Systems Engineering
- Application Domain
- Military
- Civil
- System-of-Systems Type
- Decentralized System-of-Systems
- Virtual
- Collaborative
- Centralized System-of-Systems
- Directed
- Acknowledged
- Central Authority
- Environment
- Quality Attribute
- Software Architecture
- System Architecture
- Dynamic Architecture
- System-of-System Characteristic
- System-Level Characteristic
- System Complexity
- System Evolution
- System Negative Emergence
- System Size
- System Variability
- Subsystem-Level Characteristic
- Subsystem Autonomy
- Subsystem Governance
- Subsystem Heterogeneity
- Subsystem Physical Distribution
- Subsystem Reuse
7. If you strongly disagree or disagree, please explain which concepts should be corrected or included in the ontology.













Relationships
In this section, please evaluate if the relationships between concepts are correctly mapped.

8. All relevant relationships related to the Systems-of-Systems concepts have been correctly mapped in the ontology.

Mark only one oval.

☐ Strongly disagree
☐ Disagree
☐ Undecided
☐ Agree
☐ Strongly agree

9. If you strongly disagree or disagree, please indicate which relationships should be corrected in the ontology.

Tick all that apply.

☐ System-of-Systems has Stakeholder of SoS
☐ System-of-Systems accomplishes Global Mission.
☐ System-of-Systems is applied to Application Domain
☐ System-of-Systems is classified as System-of-Systems Type
☐ System-of-Systems presents System-of-Systems Characteristic
☐ System-of-Systems realizes Mediator
☐ System-of-Systems is situated in Environment
☐ System-of-Systems satisfies Quality Attribute
☐ System-of-Systems exhibits System Architecture
☐ Constituent has Stakeholder of Constituent
☐ Constituent accomplishes Individual Mission
☐ Constituent participates in System-of-Systems
☐ Mediator mediates Constituent
☐ System-of-Systems Engineering develops System-of-Systems
☐ Stakeholder is interested in Mission
☐ Stakeholder of Constituent is interested in Individual Mission
☐ Stakeholder of SoS is interested in Global Mission
☐ Centralized System-of-Systems is managed by Central Authority
☐ System Architecture enables Quality Attribute
☐ System Evolution leads to Dynamic Architecture
☐ Subsystem Governance defines System-of-Systems Type
10. If you strongly disagree or disagree, please explain which relationships should be corrected or included in the ontology.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

Additional Information
In this section, please provide your opinion regarding OntoSoS.

11. In your opinion, is an ontology for Systems-of-Systems necessary? *

Mark only one oval.

☐ Yes
☐ No

12. If not, please elaborate your answer.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

13. Do you consider using OntoSoS in your next project? *

Mark only one oval.

☐ Yes
☐ No

14. If yes, please explain how do you plan to use it.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

15. Please use this space for any additional comments or suggestions in regards to OntoSoS.

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________