Establishment of a Quality Model for Systems-of-Systems

Daniel Soares Santos*, Brauner R. N. Oliveira*, Adolfo Duran†, and Elisa Yumi Nakagawa*
*ICMC, Department of Computer Systems
University of São Paulo, São Carlos, SP, Brazil.
†DCC, Department of Computer Science
Fraunhofer Project Center
Federal University of Bahia, Salvador, BA, Brazil.
{danielss, brauner}@usp.br, adolfo@ufba.br, elisa@icmc.usp.br

***Abstract*—Currently, Systems-of-Systems (SoS) have performed an important role in diverse application domains, with representative examples in airport, military, and smart-cities, including crisis/emergency management. SoS refer to complex software-intensive systems, resulted from interoperability of independent constituent systems, performing new missions that could not be performed by any constituents working separately. For these critical systems, their quality is undoubtedly essential. However, in general, there is a lack of studies that discuss how quality has been addressed in such systems. The main contribution of this paper is to present an experience of establishing a quality model (i.e., a set of quality characteristics/attributes, sub-characteristics, and metrics) for SoS, in particular, for the crisis/emergent management domain. This quality model has also proved to be an important support to evaluate a system of this domain. Experience such as presented in this work could be repeated in other domains, contributing to improve the quality of a diversity of critical, complex SoS that are currently being built.**

I. Introduction

Software-intensive systems have become increasingly large and complex and even essential to the whole society. These systems are sometimes resulting from interoperability of constituent systems that work together to provide more complex missions that could not be completed by any of these systems separately [1]. This new class of software-intensive systems has been referred as System-of-Systems (SoS) and can be found in different application domains, including medicine, airport, robotics, avionics, healthcare, and automotive [2], [3]. Currently, the development of SoS still presents great challenges for the classical software engineering [4], as they presents a set of unique characteristics. Moreover, these systems must assure high level of quality considering their use in diverse critical application domains.

In another perspective, software quality has been a research topic widely researched over the last three decades [5]. In this context, a well-accepted way to support quality control is to adopt software quality models. A quality model intends to make the software quality better understandable and manageable. A widely known model is the ISO/IEC 25010 quality model that has become an international standard for evaluating quality of modern software-intensive systems [6]. ISO/IEC 25010 is based on the fact that the software product quality can be specified and evaluated using a hierarchical structure of quality attributes/characteristics (e.g., reliability and performance), sub-characteristics (e.g., availability and adaptability), and metrics to measure these characteristics and sub-characteristics [7]. Specifically for the context of SoS, in spite of the great necessity of dealing with software quality during their development and evolution, there are not quality models for SoS that can contribute to control and improve the quality of these systems.

Motivated by this scenario, the main contribution of this paper is to present an experience in establishing a quality model for SoS, more specifically, for SoS in the crisis/emergency management domain. This quality model has been built in the context of a large international research project — the RESCUER project[1] — which has as a main goal to develop an SoS for that domain. To evaluate our quality model, we get opinion of specialists in software quality as well as specialists in the domain of crisis/emergency management. Additionally, we conducted an evaluation of an SoS of that domain using this quality model. Results achieved until now show the valorous role of a quality model to improve quality of SoS.

The remainder of this paper is organized as follows. Section II presents the background on SoS and quality models. Section III presents the establishment of our quality model. Section IV presents the evaluation of this model. Section V discusses on learned lessons. Finally, Section VI presents our conclusions and future work.

II. Background

This section presents important concepts related to SoS and quality models, aiming a better understanding about the topics covered in this paper.

Regarding to SoS, definitions and characterization of SoS have been increasing discussed and widespread in recent

[1]http://www.rescuer-project.org/

years. Despite the number of different definitions existing in the literature, there is still no universally recognized definition for SoS, and then, their characterization depends often on the viewpoint and system's context. In general terms, an SoS can be seen as a "supersystem" composed by complex and operationally independent systems working together to achieve higher goal [8]. According to Maier [1], an SoS can be identified and differentiated from monolithic systems by the presence of features such as: (i) Operational Independence: constituent systems are operationally independent and have their own goals, even when disconnected from the SoS; (ii) Managerial Independence: it means that each constituent system is developed and maintained by different organizations, with their own stakeholders, development teams, processes and resources; (iii) Evolutionary Development: each constituent system evolves independently and, therefore, the SoS must also evolve, where structures, functions, and purposes are added, removed, and modified; (iv) Emergent Behavior: it means that a new behavior that can not be provided by any constituent system working separately emerges; and (v) Geographical Distribution: constituent systems may be located in different places changing information among them.

SoS started to gain popularity mainly on military systems as a strategy to reach goals or deliver unique capabilities wherein a collaborative work of complex systems is needed [1], [4]. Furthermore, SoS is migrating from traditional military domains to civil domains, such as smart homes, healthcare, crisis/emergency management, and several others. In particular, an SoS in the context of crisis/emergency management allows more efficient response to crisis and incidents through integration of police, firefighters, military, and medical systems.

Achieving quality in these systems is a quite difficult task, mainly because constituent systems are sometimes developed and maintained by different organizations, with their own stakeholders, development teams, processes, and resources [9], [10]. In this context, quality models could be used to identify relevant quality characteristics that can be further used to guide the development, evolution, and evaluation of these systems [6].

A software quality model may support a better understanding about what quality is in the context of software systems, supporting diverse activities throughout system development cycle. This is done through the identification of quality characteristics that are exhibited by software systems and can be aggregated to compose the overall software quality concept. These characteristics are generally called quality attributes (e.g., maintainability, performance, and security), which are presented by quality models to define, assess and/or predict software quality [5]. The first models emerged in the early days of the software engineering area and since then, quality models are still subject of research. The first models were proposed by McCall [11], Boehm [12], and Grady [13] in, 1977, 1978, and 1987, respectively, and influenced the creation of the international standard ISO/IEC 9126-1 [14] in 2001, which was revised and replaced by the ISO/IEC 25010 [7] in 2011.

The ISO/IEC 25010 standard provides two quality models: (i) a quality in use model that provides five quality characteristics concerning software under operation by its stakeholders and (ii) a product quality model that is composed of eight quality characteristics concerning the software system apart of its stakeholders. Both are supposed to be applied to any kind of computer system that includes a software product. Characteristics of both models are decomposed into sub-characteristics that can be measured. When every sub-characteristic is measured, it is considered that the characteristic is also measured by aggregation. Having every characteristic measured, the overall quality of the product is determined. In order to achieve this goal, one or more metrics are defined and applied to each sub-characteristic, resulting in a value that represents the degree to which it is present in the final product. ISO 9126-2 [15] is an example of standard that presents metrics for measuring sub-characteristics, and may be used together with ISO 25010 to evaluate quality of a software product.

III. Establishing the Quality Model

Before present our quality model, we present the context where this model is being established and used. In the context of crisis/emergency management, the main challenge for an Emergency Command and Control Centre is to quickly obtain contextual information to answer an emergency and ensure the correct decisions. An appropriate response is essential to attenuate the occurrence of physical injuries as well as the negative outcome to the public image of the involved organizations. Decisions based on incorrect or late information have a great potential for causing more damage.

In parallel, the everyday use of mobile devices, such as tablets and smartphones, provides an enabling technology for building new software solutions for also the emergency domain. Exploring such devices as a communication tool, the RESCUER research project proposes the development of an interoperable computer-based solution to provide Command and Control Centres with real-time contextual information related to the emergency situation in industrial areas and in large-scale events. This solution relies on the collection, combination, and aggregation of crowdsourcing information mashed up with open data. Moreover, it also supports tailored announcements about the emergencies according to a particular audience (e.g., authorities, press, affected community, and general public). The RESCUER solution comprises four main constituents systems:

- Mobile Crowdsourcing Solution (**MCS**) implements suitable context-sensitive mechanisms for eyewitnesses and operational forces carrying mobile devices to provide the Command and Control Centre with information about emergency situations, taken into

consideration the behavior of people under stress situations. The users provide reports of the incidents with text, photos, and videos. Besides, the RESCUER application is able to send relevant information without user interaction. The gathered data yield a better overview of the incident location helping to decrease the time needed by the Command and Control Centre to realize the impact of an emergency;

- Data Analysis Solution (**DAS**) is composed by algorithms that process and filter the received data (e.g., image, text, and video) to extract relevant and consolidated information. This system is responsible for fusing similar data coming from different eye-witnesses, as well as analysing photos, videos, and text messages in order to extract information such as the type of incident, the position and dimensions of the affected area, people density, surrounding sources of further danger, evacuation routes, and possible approach routes for the first responders;
- Emergency Response Toolkit (**ERTK**) provides the Command and Control Centre with updated and relevant information, in an adequate format, to support decision-making during an emergency. It applies a set of solutions to manage the analyzed data coming from the DAS and presents them in a Real Time Dashboard, using adequate visualization means; and
- Communication Infrastructure (**COM**): supports the information flow between stakeholders even when traditional communication infrastructure is overloaded, by establishing Ad Hoc network communication to propagate data between users' phones and the command centre.

The RESCUER project intends to bring a clear impact and direct, innovative benefits to the society. This project defines an iterative project lifecycle, in which each subsequent iteration builds on and improves the results of the previous one. The overall strategy divides the lifecycle into three iterations steps. Basically, the iterations have been defined according to the integration of functionality (basic functions first, integration of more complex capabilities later). This facilitates the quality management, since it allows the quality evaluation of the first results and the gradual specification and maturation of the requirements of the RESCUER solution.

To support RESCUER solution development, a quality model based on the international standard ISO/IEC 25010:2011 [7] has been established. This quality model determines quality characteristics and sub-characteristics to be considered during the three iterations of RESCUER project as well as a set of quality metrics to measure each quality characteristic. In the next sections, the establishment of the quality models is detailed concerning the quality attributes selection and quality metrics definition.

A. Quality Attributes Selection

To determine which quality characteristics and sub-characteristics are relevant to the RESCUER solution, all its non-functional requirements as well as the project goals and scope were carefully analyzed. This analysis allowed to translate each non-functional requirement into ISO/IEC 25010:2011 quality characteristics, taking into account the Product Quality and Quality in Use models. In this activity, the requirement team, developers, task leaders, and project coordinators were mobilized in order to assure that all established quality attributes are appropriate and relevant regarding the RESCUER solution requirements. This involvement was very important, since, in the current phase of the project, the requirements about the RESCUER solution are still being detailed. Therefore, some quality attributes can still not be directly translated from the RESCUER requirements. Moreover, this strategy allows to obtain a consensus about all elements that compose the proposed quality model, besides to assure that the main decisions about the quality model were coherent with the systems requirements and project goals.

On the order hand, it is important to highlight that not all quality characteristics and sub-characteristics are relevant for all RESCUER constituents systems. Depending on the use purpose of the quality model (system specification or evaluation), and the constituent system considered, a different subsets of characteristics/sub-characteristics can be chosen accordingly to the specific goals and objectives. In addition, as RESCUER is an ongoing research project, its requirements can be modified during the development of the project, and thus the quality model can be revised and modified. This means that other quality characteristics that were not considered in the first project iteration, such as performance and security, can be added in the following project iterations. Figure 1 presents the quality characteristics and sub-characteristics selected to compose the quality model proposed regarding the Software Product Quality and Quality in Use aspects.

B. Metrics Definition

As early mentioned, quality metrics are used to measure the quality of a software product by measuring its quality attributes. When applying a quality metric, it is possible to obtain a quantitative value that characterizes the degree of compliance of the software to the corresponding quality characteristic. In this sense, for each quality sub-characteristics defined in the quality model, a set of appropriated metrics for their measurement was established. These metrics were selected and adapted from the international standards ISO/IEC 9126-2 - External Metrics [15] and ISO/IEC 9126-4 – Quality In Use Metrics [16].

External metrics are used to measure the quality of the software product by measuring the behavior of the system. According to ISO/IEC 9126 [15], these metrics are used during the testing stages or during the operation of the system. The measurement is performed when executing the

Quality Character.	Quality Sub Character.	Metric	Purpose of the metric	Method of application	Interpretation of measured value	Artifact or Data Source
Usability	Learnability	MUS3	What proportion of users can operate successfully a function without a demonstration or tutorial?	Number of users that adequately operated the functions by total number of users	The closer to 1.0, the better	User test, interview or user behavior observation
		MUS4	What proportion of user can operate successfully a function after a demonstration or tutorial?	Number of users that adequately operated the functions by total number of users	The closer to 1.0, the better	User test, interview or user behavior observation

Table I: Examples of Metrics

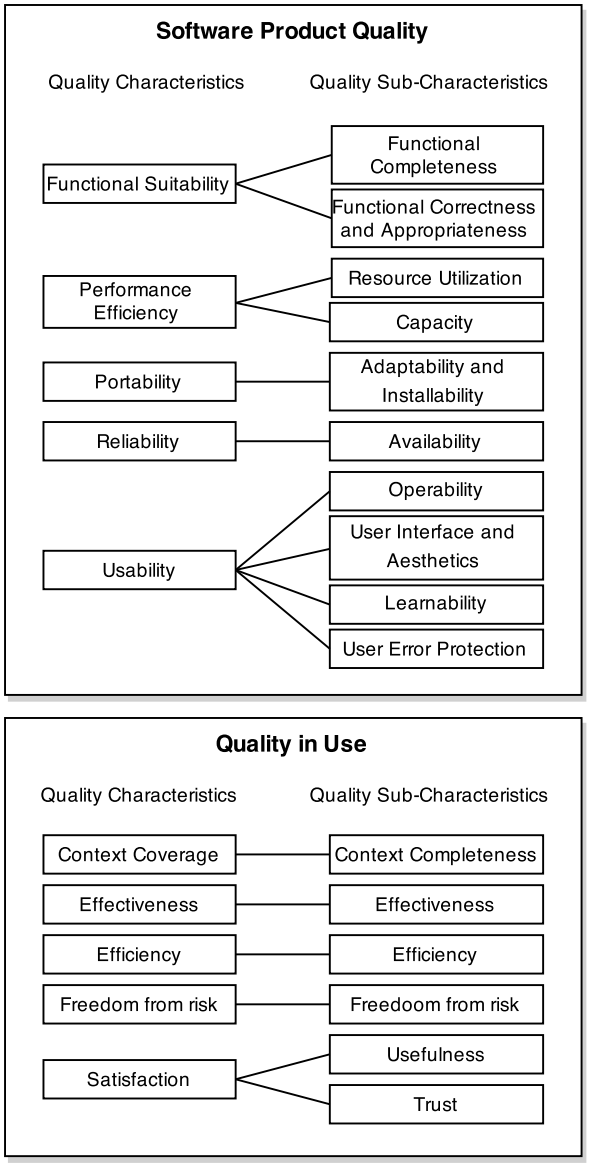

Figure 1: Established Quality Model

software product in the system environment in which it is intended to operate. On the order hand, the quality in use metrics are only applied in a realistic system environment. They measure whether a product meets the needs of specified users to achieve specified goals with effectiveness, productivity, safety, and satisfaction in a given context of use [16].

Table I presents examples of two metrics (MUS3 and MUS4) of a total of 20 metrics that we have established in our quality model. MUS3 and MUS4 were used to measure the learnability of MCS, a key quality characteristic since no training material should be necessary for the user to understand and interact with the RESCUER system during an emergency incident, even when users are under high stress situations. In this sense, these metrics are important to identify the influence of the demonstration or tutorial in the effectiveness of the users and consequently measure the level of learnability of the RESCUER solution. In addition, it is very important to highlight that specific input data are needed for the adequate application of these metrics. Input data can be obtained by using questionnaires, checklists, experiments, observations, etc. For each established metric, the method of application and the source of data or artifacts that could be used in the measurement were established.

IV. Evaluation of the Quality Model

Our quality model was validated through a survey with experts and a case study where this model was used to evaluate two different constituent systems.

The survey was performed through the application of questionnaires for 10 experts in the SoS and software quality areas to obtain evidences about the: (i) relevance of each selected quality attribute; and (ii) the applicability of the metrics and the viability of the application methods defined. Additionally, suggestions of other quality attributes that could be considered in the quality model and other metrics or application methods that could be used to measure the quality attributes were also gathered. After the questionnaires were answered, a meeting was performed with the experts in order to discuss the results and, consequently, to improve and refine the proposed quality model.

Thereafter, the quality model was used as basis to the evaluation of ERTK and MCS constituent systems. Each system was evaluated in four different situations, in Brazil and in Germany, taking into account the contexts of large events (FIFA World Cup 2014 and football games in Germany) and industrial areas (Camaçari Industrial

Quality Character.	Quality Sub Character.	Metric	First Iteration		Second Iteration		Third Iteration	
			Sub System	Acceptance Criteria	Sub System	Acceptance Criteria	Sub System	Acceptance Criteria
Usability	Learnability	MUS3	Mobile Crowdsourcing Solution	60% of the users should adequately use the app without demonstration	Mobile Crowdsourcing Solution	65% of the users should adequately use the app without demonstration	Mobile Crowdsourcing Solution	70% of the users should adequately use the app without demonstration
		MUS4	Mobile Crowdsourcing Solution	70% of the users should adequately use the app with demonstration	Mobile Crowdsourcing Solution	75% of the users should adequately use the app with demonstration	Mobile Crowdsourcing Solution	80% of the users should adequately use the app with demonstration
			Emergency Response Toolkit	N/A	Emergency Response Toolkit	50% of the users should adequately use the app with demonstration	Emergency Response Toolkit	75% of the users should adequately use the app with demonstration

Table II: Evaluation Plan

Complex[2] in Brazil). A total of 172 people participated of our evaluation. For this, a general evaluation plan was first developed in order to guide the evaluation of all constituents systems during the three iterations of the project. This general plan defines a set of assessment criteria that will be used to decide whether the metric results are satisfactory or not, considering the expected results for each project iteration. In general, these criteria are numerical thresholds or targets used to determine the need for action or further investigation. This allowed us to identify and, therefore, react in a straightforward manner to the problems that influenced the overall quality of the system. These criteria were defined through detailed analysis of the RESCUER quality requirements and refined by the requirements team, task leaders, and project coordinators.

To better manage and control the quality evolution of the RESCUER system, a different set of assessment criteria was established for each iteration. The assessment criteria were defined considering an increased level of rigor, since the metric results must improve in the course of the iterations in order to achieve the quality requirements expected to be in the final of the project. Table II presents the assessment criteria defined for the metrics MSU3 and MUS4. This table shows the increasing of the rigor level of each assessment criteria during the three iterations.

For each constituent system, it was developed a specific evaluation plan to define a strategy to obtain the needed input information for the application of the quality metrics, and, therefore, to obtain the final result about the compliance of the RESCUER constituent systems with the quality attributes established in the quality model. Since the evaluation focus of the first iteration was the usability and user experience, in each them, basically, participants were asked to use the system following a set of key tasks while their behavior was observed to identify if each task was performed following an expected way. In addition, a brief user interview was performed to identify the system acceptance and aspects regarding the user experience.

In general, the evaluation and use of the quality model can be summarized as following: (i) firstly, from the proposed quality model, it was selected the quality attributes and metrics established for each constituent systems and for the first project iteration; (ii) secondly, each quality characteristic/sub-characteristic was measured through the application of the metrics; and (iii) finally, the results were compared with the assessment criteria to identify if the quality characteristics were achieved and, consequently, to act in the quality characteristics that have not been sufficiently achieved.

Table III presents the evaluation results regarding the quality attributes defined for MCS in the first iteration of the project. As it can be observed, all the results were satisfactory considering our expectations for the first evaluation iteration of the RESCUER project. This means that, taking into consideration the average in all evaluation places, results were higher than the values of the assessment criteria. It is observed that the utilization of the quality model facilitated the identification of factors that can impact specific quality attributes of the system, as well as the quality of RESCUER solution as a whole. Through this first evaluation iteration and the feedback provided, it was observed that the RESCUER solution can be refined to achieve a higher quality for the next evaluation iteration and that there is still room for improvement in order to make the solution as intuitive as possible.

V. Learned Lessons and Discussions

The establishment and use of a quality model in the SoS context impose several challenges and difficulties mainly due to the SoS characteristics, such as managerial independence, evolutionary development, and geographical distribution. In the RESCUER project, these characteristics have proven to significantly impact the productivity, success, and effort required for the establishment of a quality model. In addition, the current quality models such as the ISO/IEC 25010 have several limitations that difficult its application in the SoS context, mainly because of the lack of clearly decomposition criteria that determine how the quality attributes achieved in the constituent systems can impact and determine the SoS quality as a whole [17]. This is a very complex problem, since quality attributes not achieved in one of their constituents can impact on the quality of other constituent systems. In addition, this impact depends on the role and importance that each constituent system plays in an SoS.

[2]http://www.coficpolo.com.br/

Quality Character.	Quality Sub Character.	Metric	Evaluation Place 1	Evaluation Place 2	Evaluation Place 3	Evaluation Place 4	Total Measure	Assessment Criteria	Total Result
Product Quality Metrics									
Usability	User Interface Aesthetics	MUS2	0.84	0.87	0.81	0.87	0.84	0.70	Yes
	Learnability	MUS3	0.39	0.58	0.73	0.70	0.60	0.60	Yes
		MUS4	0.64	0.80	0.76	0.80	0.75	0.70	Yes
Quality in Use Metrics									
Effectiveness		QUES1	0.54	0.69	0.75	0.76	0.69	0.55	Yes
Satisfaction	Usefulness	QUUS1	0.97	1.0	0.84	0.97	0.95	0.60	Yes
	Trust	QUTR1	0.97	0.85	0.78	0.93	0.88	0.60	Yes

Table III: Evaluation Results of the Mobile Crowdsourcing Solution

On the order hand, it was observed that the establishment of domain-specific quality models (from a set of quality attributes early identified by a general quality model such as the ISO/IEC 25010) must be performed with attention. There are, for instance, quality attributes important for such domains, but not present in ISO/IEC 25010. In particular, for SoS, the coverage rate of the ISO/IEC 25010 is only 44% regarding the quality attributes important for this class of systems [17]. This may significantly compromise the completeness and comprehensiveness of the quality model that was developed. Moreover, some well-established definitions for each quality attribute can not be fully applied in the SoS context due to the flexible, dynamic nature of these systems. Therefore, it is possible that some quality attributes considered in the proposed quality model can not directly express the required characteristics for RESCUER project.

In addition, considering RESCUER solution as an SoS, during the refinement of our quality model in the next two iterations, other important quality attributes, such as interoperability, security, reliability, and performance, including those ones identified in [17], will be considered. As a consequence, we intend our quality model can adequately guide the development and evaluation of the RESCUER solution and, in some extend, to serve as a reference of other quality models for other critical domains where SoS have been applied.

VI. Conclusions

SoS is becoming increasingly important and being applied in several critical sectors of the society. By their criticality, evaluation of their quality is essential. In this scenario, this paper presented an experience of establishing a quality model for SoS, in particular, for the domain of crisis/emergency management. This model was evaluated through a survey involving experts in SoS. Besides that, we also applied this model in a case study to evaluate an SoS of such domain. As a result, we observed that quality models must be adopted as one of the main guidelines to support the improvement of quality of software-intensive systems, including SoS. For the future work, we intend to apply this quality model in other evaluation iterations, as well as to update it to be consolidate as a model to be adopted for this critical, essential application domain. Besides that, we intend our experience can be reproduced in other critical domains where SoS are found.

Acknowledgments

This work is supported by Brazilian funding agencies FAPESP (Grant: 2014/02244-7), CNPq (Grant: 490084/2013-3).

References

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

[2] Elisa Y. Nakagawa, Marcelo Gonçalves, Milena Guessi, Lucas B. R. Oliveira, and Flavio Oquendo. The State of the Art and Future Perspectives in Systems of Systems Software Architectures. In *SESoS*, pages 13–20, Montpellier, France, 2013.

[3] J. A. Lane. What is a system of systems and why should i care?, 2013.

[4] Department of Defense. Dodaf architecture framework version 2.02. [On-line], 2010. (Accessed 20/03/2015).

[5] S. Wagner. *Software Product Quality Control*. Springer, Berlin, Heidelberg, 2013.

[6] N. Azizian, T. Mazzuchi, S. Sarkani, and D. F. Rico. A Framework for Evaluating Technology Readiness, System Quality, and Program Performance of U.S. DoD Acquisitions. *Syst. Eng.*, 14(4):410–426, 2011.

[7] ISO/IEC. ISO/IEC 25010 - Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models. Technical report, 2011.

[8] M. Jamshidi. *Systems of Systems Engineering: Principles and Applications*. Taylor & Francis, 2008.

[9] M. Gagliardi, W. Wood, J. Klein, and J. Morley. A uniform approach for system of systems architecture evaluation. *CrossTalk*, 22(3-4):12–15, 2009.

[10] D. S. Santos, B. Oliveira, M. Guessi, F. Oquendo, M. Delamaro, and E. Y. Nakagawa. Towards the evaluation od system of systems softaware architecture. In *WDES*, pages 1–6, Maceio, Brasil, 2014.

[11] J McCall. *Factors in Software Quality: Preliminary Handbook on Software Quality for an Acquisiton Manager*, volume 1-3. General Electric, 1977.

[12] B. W. Boehm, J. R. Brown, J. R. Kaspar, M. L. Lipow, and G. MacCleod. Characteristics of software quality. *Elsevier*, 1978.

[13] Robert B. Grady and Deborah L. Caswell. *Software Metrics: Establishing a Company-wide Program*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1987.

[14] ISO/IEC. ISO/IEC 9126. Software engineering – Product quality. Technical report, 2001.

[15] ISO/IEC. ISO/IEC 9126 - Software engineering - Product quality - Part 2: External metrics. Technical report, 2003.

[16] ISO/IEC. ISO/IEC 9126 - Software engineering - Product quality - Part 4: Quality in Use metrics. Technical report, 2003.

[17] D. S. Santos, T. Bianchi, K. R. Felizardo, and E. Y. Nakagawa. An investigation on quality attributes of systems-of-systems. Technical report, São Paulo, Brazil, 2015.