Introducing Meta-Requirements for Describing System of Systems

Introducing Meta-Requirements for Describing System of Systems

Andrea Ceccarelli, Marco Mori, Paolo Lollini, Andrea Bondavalli
Department of Mathematics and Informatics, University of Florence, Italy
Viale Morgagni 65, 50134, Florence, Italy
{andrea.ceccarelli, ma.mori, lollini, bondavalli}@unifi.it

***Abstract*- Complex, evolutionary systems operating in an open world can be seen as a composition of components which interact each other in order to fulfill their requirements. Following this vision, Systems of Systems (SoSs) literature aims at supporting the life of such complex systems taking into account key viewpoints such as emergence, time, mobility, evolution, dynamicity. Although different attempts can be found in the literature to address mostly specific viewpoints separately, it is still missing a unifying approach to analyze the whole set of viewpoints and their relationships, based on the identification of meta-requirements that can be exploited to describe any System of Systems (SoS). To this end, we developed a unifying meta-requirements model to describe SoSs viewpoints and relate them. The model is meant to be used to support the derivation of the requirements for any SoS. This paper introduces the problem, and presents the main notions of the meta-requirements model with the support of a domain-specific scenario.**

Keywords—System of Systems; RUMI; emergence; dynamicity; evolution; requirement model; AMADEOS

I. INTRODUCTION

In modern large-scale complex systems, the overall behavior of the system is the result of many, heterogeneous, autonomous, *Constituent Systems* (CSs, [1]), having complex interactions and dependencies that give birth at a complex, interacting and evolutionary *System of Systems* (SoS, [1]). Guaranteeing strict non-functional requirements in such context, e.g., performance, timeliness, dependability and safety is a great challenge and requires the combined application of several design and evaluation techniques [8].

In order to achieve a high level of assurance for such systems it is necessary to support their complete life, ranging from design to development and evolution. To this end, a set of critical viewpoints have to be considered during the life-time of an SoS. Based on the ongoing activities and outcomes achieved in the context of the AMADEOS project[1] [7], ten viewpoints have been selected and explored, which are the following (also expanded in Section II) [2], [6]: i) SoS *constraints*, that constitute the limitations and boundaries of an SoS; ii) *Architecture and Relied Upon Message Interface* (RUMI) which describes the CSs composing an SoS, and their interoperability rules as dependencies and interaction rules; iii) *semantic of communication*, which looks at the necessary preconditions in order that two legacy systems, developed by different organization using differing architectural styles, can communicate; iv) *dynamicity*, which refers to the frequent changes that may happen in the SoS, and which may have many effects on the SoS; v) *evolution*, which involves long-term changes e.g., to adapt to new standards and safety regulations, or new legislations (often called SoS evolvability), varying the intra-domain interactions and communication as well as boundaries of the SoS; vi) *emergence* (positive, negative or neutral), which is an intrinsic property of the SoS and concerns novel phenomena that manifest at the SoS level but are not observable at lower level i.e., at the level of the subsystems; vii) *governance*, which describes the way in which an SoS is managed, including defining roles, laws, standards; viii) *handling of time*, a fundamental viewpoint as SoSs are sensitive to the progression of time; ix) *dependability* and *security* (including trust and privacy), that are amongst the main concerns in critical SoSs; and ultimately x) *quality metrics*, which may be related to different aspects of an SoS and set qualitative and quantitative metrics to decide on the efficiency and effectiveness of an SoS.

Most of the above viewpoints are usually not included in "traditional" system engineering, thus calling for new approaches and methods for the design, implementation and evolution of SoSs. In order to define such approaches, it is first necessary to define the common characteristics of SoSs in terms of meta-requirements that precisely conceptualize the key viewpoints.

In this paper we present a meta-requirements model that can be applied to describe a generic SoS and to support its design, development and evolution. This model entails entities, peculiarities and characteristics that should be identified when describing an SoS, or that a designer can rely upon when writing requirements for a specific SoS. The model is intended to be generic to any SoS, and organized based on different perspectives, guided by the viewpoints.

To the best of our knowledge there are no approaches to define meta-requirements of an SoS in general. The meta-requirements are at the basis of the work carried out in the

[1] The objective of the AMADEOS FP7 project is to bring time awareness and evolution into the design of System of Systems (SoS), to establish a sound conceptual model, a generic architectural framework and a design methodology, supported by prototype tools, for the modeling, development and evolution of time-sensitive SoSs with possible emergent behaviors.

AMADEOS project, especially as input for the definition of an SoS conceptual model and a generic architectural framework that shall be carried out starting from such results. In this context, as part of the project it was developed a detailed analysis on how to obtain a SoS *meta-requirements model* [2], which is the main contribution of this paper. The work consisted in an analysis of the state of the art of SoSs followed by the study of SoSs in different domains (railway, automotive, smart energy grids, global automated teller machine network, and crisis management). Based on the analysis of the commonalities among the domains, a meta-requirements model has been synthesized.

The rest of the paper is organized as follows. Section II presents SoS basics and defines the viewpoints, motivating their selection. Section III illustrates the railway domain as a representative SoS, that will be used to discuss the meta-requirements model in later Sections. Section IV introduces the meta-requirements model, and Section V details it for the different viewpoints correlated with examples from the sample domain. Section VI concludes the paper with a discussion on open challenges for SoS engineering.

II. Basics on SoS and Viewpoints

SoS is an area of computer science which is more and more relevant since monolithic system designs have been progressively replaced by Off-The-Shelf (OTS) approaches enabling the interconnection and integration of third-party constituent systems. Different attempts in defining an SoS can be found in the literature; among those definitions the following one seems to be accepted by a majority of the SoS community [1]:
"An SoS is an integration of a finite number of constituent systems which are independent and operable, and which are networked together for a period of time to achieve a certain higher goal." An SoS comes about by the integration of *Constituent Systems* (CSs) e.g., existing legacy systems that might belong to different organizations and newly developed CSs. An SoS is composed of hardware/software systems, communication systems, physical machines and humans [2]. The most well-known classification of SoS [38] proposed four different categories, i.e., *directed*, *acknowledged*, *collaborative* and *virtual*, each providing a different degree of control and coordination. A *directed* SoS is managed by a central authority providing a clear objective to which each CS is subordinate; an *acknowledged* SoS has a clear objective but the CSs might be under their own control; a *collaborative* SoS has no clear objective and its CSs act together to address shared common interests; finally a *virtual* SoS has no clear objective and its CSs do not even know one another.

In light of the key features presented in the literature, we analyze the life of an SoS following a set of viewpoints, which aim at representing different angles of analysis for the SoS and that in our analysis include the traditional system engineering views and complement them with main SoS specificities. The viewpoints we identified are the above mentioned: constraints, architecture and RUMI, semantic of communication, dynamicity, evolution, emergence, governance, handling of time, dependability and security, quality metrics. We introduce them in the following.

Constraints can be defined as the conditions that should be respected in order to achieve a given goal state following a certain path [6]. Constraints represent different sources of restrictions related to standards, assets, economic aspects, technological solutions, etc.

The *architecture* of an SoS can be defined as the manner in which CSs are organized and integrated [34]. It defines the structure of composition and the behavior. A relevant related argument is the *RUMI*, that is a message interface for the exchange of information among two or more CSs that establishes a well-defined boundary between the CSs that forms part of a backbone of an SoS system architecture [6]. It represents the support for the exchange of information among two or more CSs through well-defined boundaries.

Semantic of communication refers to the meaning of the information exchanged between different CSs [37], [22]. It describes how and why the information is exchanged among the CSs (usually through a RUMI [35]). Note that while the Architecture and RUMI viewpoint includes the syntactical aspects of messages and protocols, this viewpoint refers to the semantics of exchanged information.

SoSs must cope with short-term and long-term adaptations. We refer to the former as *dynamicity*, and to the latter as *evolution*. Dynamicity is the property of an entity that is constantly changing in terms of offered services, built-in structure, and interactions with other entities [6]. It aims at reconfiguring the SoS in specific situations, e.g., either after a fault or after the variation of an external condition. Evolution refers to long-term and continuous changes in the SoS as new functions are included, removed or modified according to the evolution of needs [8]. It aims at reconfiguring the SoS in face of changing requirements, e.g., new business requirements.

Another important characteristic of an SoS is the so called *emergence* [6], [13], [22] that is defined as a phenomenon at the macro-level which is new with respect to the non-relational phenomena of any of its proper parts at the micro level [6]. The rationale behind emergence is that the composition of CSs may lead to global *emergent phenomena*, either positive or detrimental. Managing emergence, an SoS will be able to avoid un-safe unexpected situations generated from safe CSs and it will be able to profit from positive emerging phenomena. Examples of emergence in the ICT world can be found in [13].

Governance is the theoretical concept referring to the actions and processes by which stable practices and organizations arise and persist [36], [6]. It describes the way in which an SoS is managed, including defining roles, laws, standards, etc. This is a critical aspect because different owners/regulations related to different CSs may limit the interoperability across them. Governance can be also considered as a part of SoSs' constraints.

Since an SoS is sensitive to the progression of time, it is needed to reliably *handle time* throughout the SoS life [22]. While in a monolithic system a common clock for time-stamping events can be derived directly from the signals of the

central physical oscillator, no such common clock exists in an SoS. The establishment of a global time based in a distributed system is extensively discussed in [20].

Dependability (the ability to avoid failures that are more frequent and more severe than it is acceptable, [14]) and *security* are a fundamental viewpoint for the success of critical SoSs [32], [33]. Dependability and security deal with non-functional critical requirements as for example system availability, reliability, safety, privacy or confidentiality.

Quality metrics refer to the set of qualitative and quantitative metrics to decide on the efficiency and effectiveness of an SoS. They have been applied to different concerns, among others software quality [29], grid computing [30], cloud computing [31], social networks [40]. Quality metrics is a transversal viewpoints which may embrace several of the above viewpoints. Quality metrics are related to the many different aspects of an SoS and they are often at the basis of SoS constraints (e.g., Quality of Service metrics [40]).

To the best of our knowledge no approaches consider all the above viewpoints together in conceptualizing an SoS. A few approaches entails just a few of the viewpoints but still they do not take into account relevant key elements such as handling of time, governance and quality metrics. Among others, the approach presented in [3] introduces five different viewpoints; *i) operational independence; ii) geographical distribution; iii) emergent behavior; iv) evolutionary/adaptive development.* The approach presented in [4] is another attempt in considering different SoS viewpoints jointly in a single framework. It proposed as basic viewpoints *interdisciplinary*, *heterogeneity of CSs* and *networks of systems*. Finally the approach presented in [5] by Boardman and Sauser introduced *Autonomy*, *Connectivity*, *Diversity*, and *Emergence* as key viewpoints for SoS. It is worth noticing that the viewpoints presented in [3], [4], [5] even if they have been called with different names, they still can be easily mapped to our proposed viewpoints.

III. Rationale and Approach

As building a meta-requirements model applicable to any SoS is a challenging and non-conventional exercise, we are aware that different approaches may be adopted and questioned. Thus the objective of this section is to illustrate the rationale we adopted, presenting the approach and structure for the meta-requirements of a generic SoS.

Our SoS meta-requirements are intended to be SoS elements, peculiarities, or characteristics that should be identified when describing or designing an SoS.

The definition of the meta-requirements was guided by an analysis on SoS domains carried out in [2] and enlisted in the Section I. An analysis of cross-domain commonalities and differences was performed, and its results guided us towards the compilation of the meta-requirements for a generic SoS.

The resulting model adopts the ten viewpoints presented in Section II, and it identifies relations between them. For each viewpoint, a block of requirements is identified, and structured in sub-views (sub-blocks of requirements) as follows.

For the Constraints viewpoint, requirements are structured as: i) general requirements; ii) relations with other viewpoints; iii) an analysis of four identified classes of constraint (while other classes of constraints may be identified and added, we explored four that we believe are the most relevant: these are assets, standards, financial, and governance). Note that governance is a viewpoint, but it is so strongly related to constraints that it is considered also as a constraints class.

For the Architecture & RUMI viewpoint, an approach typical of system requirements definition is adopted. It consisted in grouping requirements on the basis of the five traditional views for the description of architectures following the ISO RM-ODP classification [15], i.e., i) enterprise view (defines the objects, the environment, the roles, and the activities executed in the system), ii) information view (focuses on the information exchanged), iii) computational view (as it is specifically intended for software engineers, it has little relevance in our framework), iv) engineering view (adopts the point of view of a system architect) and v) technology view (refers to the applicable technologies and standards). Differently from RM-ODP approach we only use its proposed classification to collect together related requirements while we do not detect any mutually coherent descriptions among viewpoints since we do not propose concrete artifacts but meta-requirements.

For the remaining viewpoints, the requirements are structured as: i) general requirements specific to the viewpoint with no or minimal overlaps with the other viewpoints, ii) requirements which describe how the considered viewpoint is influenced by other viewpoints, iii) requirements which describe how the considered viewpoint influences other viewpoints, iv) a graphical summary of relations with other viewpoints as resulting from ii) and iii).

Despite all our efforts, we cannot claim that we have identified a mapping of relations between viewpoints which can be considered complete and does not include unnecessary relations. Such mapping is extremely difficult to define, and some relations between viewpoints may be (and in our experience, have been) valued differently by different people. Hence, when building an SoS, it is necessary to add or remove relations between viewpoints in case it is appropriately justified by the specifics of the considered SoS.

Table 1 presents the number of requirements for each viewpoint and summarizes the relations identified. For space

Table 1. Meta-requirements per viewpoint and relations.

ID	Viewpoint	# meta-requirements	Relations with
1	constraints	20	all viewpoints
2	architecture and RUMI	21	all viewpoints
3	semantic of communication	11	1,2,4,5,6,7,9,10
4	dynamicity	7	1,2,3,6,7,8,9,10
5	evolution	8	1,2,3,6,7,10
6	emergence	8	all viewpoints
7	governance	5	all viewpoints
8	handling of time	12	1,2,4,6,7, 9,10
9	dependability, security	16 dependability 11 security	1,2,4,6,7,8,10
10	quality metrics	5	all viewpoints

constraints in Section V we show the main findings, with the help of the railway domain (presented in Section IV) to support them with evidences and examples.

IV. A Motivating Scenario: the Railway Domain

The railway system moves people and goods within a country and between countries. This system is critical to the economic and social wellbeing of several nations. In Europe, based on quarterly figures, rail passenger transport performance at EU-27 level continued to increase by around 3 billion passenger-kilometers between 2010 and 2011 [9].

All railways have the same basic targets. Beyond a safe railway, they are all working to maximize the capacity at which they can operate their networks, minimize passenger and freight delays, maximize the reliability of the infrastructure and rolling stock, and do all of these at minimum cost [10]. Since disruptions of the railway infrastructure can have a significant negative impact on the economy and security of an individual country [11] and the current railway system depends on ICT (typically to increase performance), the *security* and *safety* aspects (especially for wireless communications) become critical. Especially for passengers transportation, additional undeniable requirements of the railway infrastructure are availability and timeliness of the train transport service, as well as accessibility and functionality of train stations, together with a sufficient degree of comfort.

The railway infrastructure is able to sustain such enormous quantity of people and goods moved daily. This makes it a complex infrastructure, composed of several items, players, and dependencies with other connected infrastructures also beyond the railway domain (e.g., the Merano train accident caused by a burst irrigation pipe placed near the tracks [26]). To express the quantity of relevant interacting items of the railway infrastructure and to suggest an idea of the complexity of the railway domain, we mention (from [12]) ground areas; tracks; engineering structures; bridges, culverts and tunnels; superstructure; access way for passengers and goods, including access by road; safety, signaling and telecommunications installations on the open track, in stations and in marshalling yards; lighting installations for traffic and safety purposes; plant for transforming and carrying electric power for train haulage: sub-stations, supply cables between substations and contact wires, catenaries.

The complexity of the railway SoS, together with its classification as a major critical infrastructure, calls for the investigation of the SoS to design solutions for better, safer and securer interoperability of the different CSs and with respect to the adjacent SoSs. In Section V the railway domain is treated as an SoS, and the meta-requirements model is discussed considering examples from the selected domain.

V. Meta-Requirements per Viewpoints

A. Constraints

We present the main meta-requirements for this viewpoint. When describing an SoS, constraints *shall* be identified and defined in order to limit the solution space for an SoS, according to conditions of heterogeneous nature ranging from high level business rules till to operational constraints. Constraints *shall* be organized in classes; as a minimum, assets, standards, system life, financial, governance.

Assets. Assets are elements which are fundamental for the existence of an SoS. Assets spans from physical entities to patents and technologies. Identification of assets is non-univocal and depends on the SoS and the point of view. For example, a manager of the railway network which describes the railway SoS may identify a train as an asset; an engineer working to design a train most likely will not consider the train as an asset, because it is his final objective. Assets *must* be listed in classes, describing at least the scope, lifespan, economic value, criticality, number of instance.

Standards. An SoS and its entities *may* be subject to the mandatory or recommended application of standards. For example the EN 501xx [21] family of standards is mandatory in Europe to regulate the railway safety lifecycle development process.

System life. The system life influences and limits the development techniques, the evolution, the processes, the strategic choices, the time to deployment, the life-span and lifecycle, and finally the costs. An SoS and its CSs *shall* have their own system life described by attributes as lifecycle, process, time of life, role. System life of different SoSs or CSs *may* have dependencies, thus leading to connections between them. *It is required* to explore the system life to understand if its costs, time, and complexity are acceptable. For example, railway tracks, bridges, tunnels, etc. are built with high effort and cost, but they are supposed to last decades.

Financial. Financial constraints and costs influence the dimension and complexity of an SoS. Cost effectiveness is often required in SoS design and management; a cost sensitive approach impacts the selection of lifecycle, services, functionalities, etc. For example, a train is particularly expensive, but it can be operative for several years with the appropriate maintenance.

Governance. The dimension, complexity, services, and usage of an SoS *shall* be regulated by governance constraints. These include, but are not limited to, constraints to the players (stakeholders, users, satellite activities, funders), the technology, the evolution and progress, the financial aspects, the management of the SoS. An example in railway is the management roles of the railway SoS, composed of both public and private entities. The role of the different players (e.g., tracks are usually owned and managed by public entities), and dependencies from funding schemes (mostly comes from public entities), are example of the governance constraints of the railway domain.

Constraints are related to all viewpoints, and typically limit them. Changing constraints may lead to changes in *all* other viewpoints. For example, in the railway domain the safety regulations are defined in standards as the IEC EN50126 [21] for the electronic equipment. Changes in the standard may set new dependability and security requirements, leading to changes in the systems or the operation procedures.

B. Architecture & RUMI

The main meta-requirements identified in the *enterprise view* specify that an SoS *shall* have a type, to be selected amongst Directed, Acknowledged, Collaborative and Virtual.

The interplay of CSs *shall* create compositions, organized in hierarchical or holarchical levels to form another CS or an SoS. Thus an appropriate architecture and RUMI *shall* be defined to manage the structure of an SoS in terms of a set of CSs. Noteworthy, the RUMI is the only means for communication in SoS, and it *shall* describe *all* interactions between machines, humans and controlled objects. Limitations on the possible specifications of RUMIs are due exclusively to the constraints.

Meta-requirements of the ODP *information view* report that the RUMI *shall* describe the semantic of communication through the definition of *Itoms* (Information Atom [22] i.e., a tuple consisting of data and the associated explanation of the data), that are exchanged between the entities that compose the SoS and its boundaries.

The ODP *engineering view* mainly discusses from the viewpoint of a system architect meta-requirements on i) handling of time, dependability, security, ii) emergence. Regarding i), it is *mandatory* to consider architectural solutions for dependability, handling of time and security at SoS level. When requirements on dependability, security, handling of time, quality metrics are demanded, the SoS *shall* be observed for monitoring purpose. The notion of time *shall* be set for the SoS, defining whether it is needed a shared time base [22] and resilient time synchronization [17]. Regarding ii), the RUMI interactions of CSs and/or humans and controlled objects *shall* generate emergent phenomena at the upper level.

In the ODP *technology* and *computational views*, a remarkable meta-requirement is that limitations to techniques and technologies adopted are set by the constraints. For example, railway safety standards pose serious limitations on the acceptable programming languages (PHP, Ruby, Java, are rarely if not ever used) [21].

The railway domain is an example of acknowledged SoS. We present a possible hierarchical organization of the railway SoS according to the European ERTMS (European Rail Traffic Management System [23]), and the related RUMI.

At the top of the hierarchy, the GSM-R (GSM for Railways) system and the ETCS (European Train Control System) define interactions between the signaling items, respectively for i) information transmission between the track and the train, and ii) transmitting the permitted speed and movement information to the train driver and constant monitoring of the driver's compliance with the instructions.

The GSM-R system can be further decomposed in the GSM-R antennas, while the ETCS encompasses the railway on-board subsystems and the trackside (ground) subsystems, including the RBC (Radio Broadcast Center, in charge of elaborating information to and from the trains, that are transmitted by means of GSM-R). The RUMI is defined following the standard specifications [23].

C. Semantic of Communication

Semantic of communication *shall* be described by Itoms, and it *shall* be defined for the SoS and its boundaries. Limitations to the possible semantic of information exchanged are set only by constraints.

Semantic of communication has clear relations with constraints (e.g, in the ERTMS described above, the Itoms exchanged are limited by standards) and strong similarities and connections with architecture and RUMI. The latter could even be considered as a single viewpoint with Semantic of communication but we still treat them separately to distinguish issues related to semantic of information from issues related to the architectural infrastructure, consistently with deliverable D1.1 of AMADEOS [2]. Other relations exist with other viewpoints as presented in Table 1. For example, changes in the semantic of communication may influence dynamicity and emergence, giving birth to new compositions of CSs and functionalities at SoS level.

D. Dynamicity

In an SoS, changes *may* happen and have many different effects on the SoS. For example, changes *may* lead to new emergent phenomena. Thus the viewpoint dynamicity required a careful analysis, to understand the viewpoints that are related to dynamicity (i.e., generates dynamicity or are affected by dynamic phenomena).

The dynamicity of the SoS/CS and connected humans and controlled objects *shall be* limited in frequency, number, and dimension by its constraints and the Architecture & RUMI. Dynamicity *may* be caused instead by modifications in the Architecture & RUMI, semantic of communication and constraints.

For example, several large scale systems nowadays show trends towards self-organization and adaptiveness. The Itoms that are exchanged by these system may change abruptly, for example due to new services which are required or due to changes in the environment. Thus semantic of communication is not fixed - and consequently, changing the semantic of communication may cause dynamicity.

In general, the dynamic behaviour *may* require changes to the Architecture & RUMI and the Semantic of Communication. Dynamic behaviour *may* influence security (it is sufficient to imagine a system administrator without appropriate training that fails in keeping all the services and systems updated and properly configured), system dependability (for example, erroneous services orchestrations may reduce the Quality of Service offered to the users), and handling of time (for example abruptly increasing the workload of a web market may cause the service to slow down, thus missing deadlines). Such influence on dependability, security, and handling of time *shall* be analyzed and, whenever possible and necessary, mitigated.

We also consider dynamicity as a possible cause of emergence: changes in the dynamicity of a CS/SoS *may* give rise to emergence phenomena.

A typical approach to dynamicity in railway safety-critical system is based on reducing the potential detrimental

consequences due to the happening of hazardous events. In general, a safe state is included in the system, that prevents further operations of the equipment. Such state is entered when an hazardous event is detected which potentially could lead to an accident. A typical safe state is the shutdown of the component, which for on-board equipments may lead to a train halt. While this could be defined as a very conservative approach, e.g., with respect to adaptiveness, it is widely considered reasonable for train-borne equipment, where safety of the train mission has priority on reliability.

E. Evolution

Main meta-requirements for the evolution viewpoint are that the SoS and the CSs that compose the SoS *may* evolve, varying the intra-domain interactions and communication and potentially also the external boundaries. As evolution is an inherent property of SoSs, *it is required* to take into account evolution when building an SoS and when describing it. The limitations to evolution and its effects on the other viewpoints *must* be well understood to achieve an attentive SoS design.

Regarding relations to the other viewpoints (see also Table 1), evolution is governed *exclusively* by the constraints. Evolution of an SoS *may* give rise to modifications in the Architecture & RUMI and Semantic of Communication, and may originate Emergent phenomena.

We would like to remark that amongst the viewpoints connected to evolution, dependability, security and handling of time are missing. While short-term modifications to a CS or SoS due to dynamicity may lead to changes in the security, dependability and time handling, we believe that long-term changes due to evolution do not reflect directly on such viewpoints. Other viewpoints influenced by evolution are responsible for changes in dependability, security and handling of time. For example, evolution may lead to changes in the Architecture & RUMI which can provoke changes in security, dependability and handling of time solutions. Or as another example, evolution may lead to (detrimental) emergent phenomena that may impact dependability, security, handling of time.

The ongoing changes in the European railway governance offer a nice example of evolution. The European railway scene is now changing very rapidly, partly under EU pressure, moving towards industry fragmentation, and enhanced interoperability [24]. States are moving in the direction of separating the provision of infrastructure from train operation; there are new train operators, especially freight. The ultimate aim of these changes is the overall modernization of the railway business in order to render this industry less dependent on subsidies for its financing, along with improved flexibility and capacity to face complex environments, helping its integration into the global transport system [25].

This fragmentation requires a response to ensure the maintenance of dependability, and especially safety. There is also pressure towards increased "interoperability", that is, the movement of trains and locomotives across international boundaries, thus requiring harmonization of RUMI and semantic of communication. Also, this requires steps towards the harmonization of technical standards that will guide the evolution process.

F. Emergence

It is both fundamental and deeply challenging to predict emergent behaviors in SoS, especially detrimental ones. Appropriate effort *shall* be devoted to observe, measure and predict detrimental emergence phenomena and to mitigate their effect on the SoS. For non-detrimental emergence, it is instead positive but *not deemed mandatory* to observe, measure and predict emergence phenomena.

Emergence phenomena *may* be influenced or generated by modifications to the Architecture & RUMI (e.g., adding new components which introduces new functionalities, or adding new components that may change the error model), the constraints (e.g., introducing new assets), the semantic of communication (e.g., introducing new Itoms which enables new interoperability between CSs), dynamicity and evolution (these relations are already discussed in the dynamicity and evolution viewpoints). Note that emergence phenomena *may* cause violations to the constraints, handling of time, dependability and security of the SoS/CS.

Surveying the state of the art, we were not able to match the topic of emergence to the railway SoS. Anyway, the two following elements can be considered as potential sources of detrimental emergent phenomena. The first is the interactions between different domains or different SoSs. An example is the Merano accident where the explosion of an irrigation pipe nearby the track was at the origin of a deadly train derailment [26]. The second is the growing complexity of the railway SoS and the increasing number of CSs. The absence of clear interfaces or interoperability of the involved CSs makes very difficult to model or monitor the whole SoS, including assessing or mitigating hazards of the SoS.

G. Governance

Main meta-requirements for the governance viewpoint are reported below. A governance model *is required* in order to support the achievement of the SoS mission by means of the sub-goals of its CSs. The CSs cooperatively interact one another to achieve the SoS goal(s). Incentives *must* be provided and trust *must* be established among the CSs in order that the selected CSs are cooperative. Governance *may* change through time and it *shall* contribute to the definition of SoS boundaries.

The governance viewpoint has relations with all viewpoints. In particular, governance is both a constraint and influenced by constraints, for example for what concerns organizations and management.

Governance of the railway system offers an example of supranational critical infrastructure, with a vast range of interacting players. Considering the European railway system, the major stakeholders are: European Commission, which defines guidelines for railway system integration; Member States, which supervise the system; private/public companies, which implement and manage the infrastructure and local communities, which benefit from the service to transport goods and people. Also the list of players involves several

actors, from rail transportation companies to supply companies, employees and ultimately passengers [27].

H. Handling of time

As an SoS is sensitive to the progression of time, it *shall* have requirements describing its handling of time. These can be organized in timeliness requirements and time synchronization requirements. Every CS in the SoS that is subject to physical time requirements *shall* be able to measure time with an appropriate uncertainty, and *shall* be able to achieve a quality of time synchronization which is deemed sufficient [18], [19].

In the railway domain, several systems have hard real-time requirements, e.g., for emergency brake activation. Other components may have soft real-time requirements. In general, the number of existing components implies a relevant variety in terms of real-time requirements and deadlines to meet.

The concept of global time is not widely adopted in the railway domain. For example in the ERTMS, while time can be acquired by GPS for non-safety critical items, the execution of safety-critical functions does not rely on time and clocks.

I. Dependability and security

When describing an SoS, the dependability requirements of the SoS *shall* be identified; to be noted that humans and controlled objects that interact with a CS may have non-unique, different perceptions of dependability, and different needs. Dependability and security requirements of an SoS, and assessment and monitoring solutions (these last one, defined in the Architecture & RUMI), *may* be imposed by the constraints. Note that assessment here includes dependability assessment, security assessment, and time analysis. For example, standards may indicate what assessment activities to perform.

System life, cost, governance, technological constraints *may* impose limitations to the variety of monitoring solutions applicable in the SoS. Finally, standards constraints *may* indicate what should be mandatorily monitored.

Dependability and security requirements *may* change due to modifications in the Constraints; detrimental effects of emergence phenomena *may* influence the dependability and security of the SoS or CS.

The railway system is composed of different components and an holistic approach is required to satisfy its dependability and security requirements. Railway standards define recommended design and evaluation techniques to apply to hardware and software components to guarantee safety of electronic equipment [21]. Security in the railway systems is usually intended in terms of "issues to safety" and consequent level of protection; thus, the standard [28] presents guidelines to protect literally the "safety of communication".

J. Quality Metrics

Main meta-requirements for the quality metrics viewpoint are reported below. An SoS *should* be characterized both in qualitative and quantitative terms on how well it serves its purpose. To this aim, quality metrics *shall* be precisely defined and evaluated at the SoS and CS level. Quality metrics are related to software, hardware, the communication among CSs and to performance of CSs and SoSs, as well as non cyber-physical system-related metrics such as financial, political or economical indicators. Note that we do not introduce any meta-requirement on how to measure the identified metrics.

All viewpoints influence quality metrics. The decision regarding which quality metrics to set, and why, is strictly dependent on the requirements defined in all the other viewpoints.

Several different quality metrics can be identified in the railway domain. For example, railway safety standards [21] propose qualitative and quantitative levels for the safety of equipment, called Safety Integrity Levels or SILs; service availability and reliability are usually indicated by stakeholders; the different players of the railway domain set their own financial and economical indicators.

VI. Conclusions and Future Challenges

This paper introduced a meta-requirements model relevant for generic SoSs. Our SoS meta-requirements can be seen as SoS elements, peculiarities, or characteristics that should be identified when describing an SoS.

The several facets of SoSs as emerged from the effort in defining meta-requirements, call for the reconsideration of the traditional approaches for system engineering, searching for a change in the traditional perspective for system design, assessment, implementation, deployment and maintenance. For example, viewpoints traditionally not considered when building systems here become central (e.g., emergence and governance). Moreover, other viewpoints are intrinsic to the SoS engineering (SoSE, [1]) approach (e.g., emergence, evolution and dynamicity). This calls for an effort in building a new approach for defining SoS requirements and ultimately for SoS design and assessment. The ultimate outcome of this paper is the definition of SoS meta-requirements to support the successive development of an SoS conceptual model [6]. As for future work we also envision the definition of an architectural framework upon which more extensive qualitative and quantitative analyses will be carried out [7]. The analysis of requirements of this paper identified challenges in the perspective that is offered by SoSE:

1. Being able to *completely identify the relations between viewpoints* and their interplay for the whole life of an SoS. Note that some relations are SoS-dependent, thus relations may vary from domain to domain and from SoS to SoS.

2. *Emergence* is a critical viewpoint, which was rarely if not at all present in traditional system engineering. While vast literature on emergence in SoSs exists (e.g., [13], [16], [39]), and despite the fact that emergence is an intrinsic characteristic of SoSs, there are no SoS design approaches which are emergence-oriented or emergence-aware. Similarly, models that forecast emergence are missing nowadays. Techniques for emergence-aware design, design-for-emergence, or emergence-driven design may become relevant assets in SoS design whenever appropriately identified.

3. Management of *time*, *dependability* and *security*. Time, dependability and security requirements may become more

complex when the focus is the whole SoS. This is due to the overlaps of other viewpoints that are traditionally ignored when building systems. This consideration may affect both the design and the assessment of SoS.

4. *Dynamicity* and *evolution*. An SoS is an evolutive system and it may (and most likely, will) exhibit dynamic behaviors. Other viewpoints may influence the dynamicity and evolution; this means that the SoS adaptiveness and flexibility shall be considered from both long-term and short-term perspectives, and consequently specific engineering approaches may require revisions.

5. *Constraints* and *governance*. SoS engineers move their perspective from looking at customer requirements only to looking at a plethora of constraints. These requirements should be considered as they may impact the life and ultimately the success of an SoS. Thus, the collection of requirements requires an understanding of a potentially vast amount of scenarios, where boundaries may be difficult to set and where diverse actions and processes may be defined to govern the system. To be able to build an SoS, engineers need to understand where to set the boundaries that should be considered for their work, and how broad and detailed their view on the SoS should be.

ACKNOWLEDGMENTS

This work has been partially supported by the European Project FP7-ICT-2013-10-610535 AMADEOS (Architecture for Multi-criticality Agile Dependable Evolutionary Open System-of-Systems).

REFERENCES

[1] M. Jamshidi, "Systems of Systems Engineering—Innovations for the 21st century", Wiley and Sons, 2009.

[2] AMADEOS Consortium, "Deliverable 1.1 - SoS Commonalities and Requirements", 188 pages, June 2014. http://amadeos-project.eu/.

[3] M. W. Maier, "Architecting Principles for Systems-of-Systems", System Engineering, vol. 1, no. 4, pp. 267-284, 1998.

[4] D.A. DeLaurentis, "A taxonomy-based perspective for Systems-of-Systems design methods", IEEE International Conference on Systems, Man and Cybernetics, 2005.

[5] J. Boardman, and B. Sauser, "System of Systems-the meaning of of", IEEE/SMC International Conference on System of Systems Engineering, 2006.

[6] AMADEOS Consortium, "Deliverable 2.1 - Basic SoS concepts, glossary and preliminary conceptual model", 72 pages, June 2014. http://amadeos-project.eu/ [last accessed 10 September 2014].

[7] FP7-ICT-2013-10-610535 AMADEOS - Architecture for Multi-criticality Agile Dependable Evolutionary Open System-of-Systems - http://amadeos-project.eu/ [last accessed 10 September 2014].

[8] S. Murer, B. Bonati,B. and F.J. Furrer, "Managed Evolution – A Strategy for Very Large Information Systems", Springer, 2010.

[9] European Commission EUROSTAT, Railway freight transport statistics, http://epp.eurostat.ec.europa.eu [last accessed 10 September 2014].

[10] INTEGRAIL - Intelligent Integration of Railway Systems, Publishable Final Activity Report, IGR-P-DAP-156-07, 2010.

[11] G. Baldini et al. "An early warning system for detecting gsm-r wireless interference in the high-speed railway infrastructure", International Journal of Critical Infrastructure Protection, 3(3-4):140-156, 2010.

[12] Lasting Infrastructure Cost Benchmarking LICB - Glossary, 12 May 2004, v1.0, https://www.uic.org [last accessed 10 September 2014].

[13] J. Mogul, "Emergent (Mis)behavior vs. Complex Software Systems", EuroSys, 2006.

[14] A. Avizienis, et al. "Basic concepts and taxonomy of dependable and secure computing", Dependable and Secure Computing, IEEE Transactions on 1.1 (2004): 11-33.

[15] ITU T Rec. X.901 | ISO/IEC 10746 1: Information technology – Open distributed processing – Reference Model: Overview.

[16] J. Holland, "Emergence-from Chaos to Order", Oxford University Press, 1998.

[17] A. Bondavalli, A. Ceccarelli, and L. Falai, "Assuring resilient time synchronization", IEEE Symposium on Reliable Distributed Systems (SRDS), pp. 3-12, 2008.

[18] A. Bondavalli et al., "Resilient Estimation of Synchronization Uncertainty through Software Clocks", International Journal of Critical Computer-Based Systems (IJCCBS), Vol. 4, pp. 301 - 322, 2013.

[19] A. Bondavalli, F. Brancati, A. Ceccarelli, "Safe Estimation of Time Uncertainty of Local Clocks", Int. Symp. on Precision Clock Synchr. for Measurement, Control and Communication (ISPCS), pp. 1-6, 2009.

[20] H. Kopetz, "Real-time systems: design principles for distributed embedded applications", Springer, 2011.

[21] IEC/EN 50126-1:2012 - Railway applications - The Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS), 2012.

[22] H. Kopetz, "Conceptual Model for the Information Transfer in Systems of Systems", Proc. of ISORC 2014. Reno, Nevada. IEEE Press. 2014.

[23] UNISIG, ERTMS Users Group (2008) Subset026-5, System Requirements Specification, 2008.

[24] R. Vickerman, "High-speed rail in Europe: experience and issues for future development", The annals of regional science 31.1, pp. 21-38, 1997.

[25] IMPROVERAIL, "Deliverable 10 - Project Handbook", 2003.

[26] European Railway Agency (ERA), "Intermediate report on the development of railway safety in the European Union", 2013.

[27] TENACE, "Deliverable 1 - Threat models and attack analysis", http://www.dis.uniroma1.it/~tenace/ [last accessed 10 September 2014].

[28] EN 50159:2010 - Railway applications - Communication, signalling and processing systems - Safety-related communication in transmission systems, 2010.

[29] ISO/IEC 9126, "Software engineering - Product quality", 2001.

[30] D. Colling, et al. "On quality of service support for grid computing " Grid Enabled Remote Instrumentation, Springer US, pp. 313-327, 2009.

[31] D. Armstrong, and K. Djemame, "Towards quality of service in the cloud", Proc. of the 25th UK Performance Engineering Workshop, 2009.

[32] G. Peterson, "Service Oriented Security Architecture", Information Security Bulletin Vol. 10, pp. 325-330. November 2005.

[33] D. Alagappan, et al., "Security in Service Oriented Architecture: A Survey of Techniques", In Proceedings of WISOA, 2006.

[34] E. Y. Nakagawa, et al., "The state of the art and future perspectives in Systems-of-Systems software architectures", In SESoS'13, pp. 13-20, ACM, 2013.

[35] S. A. Selberg, M. A. Austin , "Toward an Evolutionary System-of-Systems Architecture", International Council on Systems Engineering INCOSE, 2008.

[36] S. Henson, et al., "State of the Art on Systems-of-Systems Management and Engineering", Deliverable TAREA-PU-WP2-R-LU-9, 2012.

[37] H. Kopetz, "Why a Global Time is Needed in a Dependable SoS?", Proc. of the Workshop on Engineering Dependable Systems-of-Systems, Univ. of Newcastle upon Tyne, 2014.

[38] J.S. Dahmann and K.J. Baldwin, "Understanding the Current State of US Defense Systems-of-Systems and the Implications for Systems Engineering", In Systems Conference, pp. 1-7, 2008.

[39] L. Vinerbi et al., "Emergence: A new Source of Failures in Complex Systems", In DEPEND 2010, pp. 133-138, 2010.

[40] L. Coppolino, S. D'Antonio, L. Romano, F. Aisopos, K. Tserpes, "Effective QoS Monitoring in Large Scale Social Networks", IDC 2013: 249-259.