System-of-Systems to Support Mobile Safety Critical Applications: Open Challenges and Viable Solutions

Andrea Bondavalli[1,2], Andrea Ceccarelli[1,2], Paolo Lollini[1,2], Leonardo Montecchi[1,2], Marco Mori[1,2]
[1]Università degli Studi di Firenze, Dipartimento di Matematica e Informatica, Firenze, Italy
[2]Consorzio Interuniversitario per l'Informatica (CINI), Università degli Studi di Firenze, Firenze, Italy
{bondavalli, andrea.ceccarelli, paolo.lollini, leonardo.montecchi, ma.mori}@unifi.it

***Abstract*— A dramatic shift in system complexity is occurring, bringing monolithic system designs to be progressively replaced by modular approaches. In the latest years this trend has been emphasized by the *System of Systems* (SoS) concept, in which a complex system or application is the result of the integration of many independent, autonomous *Constituent Systems* (CS), brought together in order to satisfy a global goal under certain rules of engagement. The overall behavior of the SoS, emerging from such complex interactions and dependencies, poses several threats in terms of dependability, timeliness and security, due to the challenging operating and environmental conditions caused by mobility, wireless connectivity, and the use of off-the-shelf components. Referring to our experience in mobile safety-critical applications gained from three different research projects, in this paper we illustrate the challenges and benefits posed by the adoption of an SoS approach in designing, developing and maintaining mobile safety-critical applications, and we report on some possible solutions.**

***Index Terms*— System of Systems; cyber-physical; mobile; architecture; emergence; dynamicity; evolution; time**

I. Introduction

Recent advances in technology have brought significant changes in the ways systems are designed, developed, and maintained. As technology has advanced, more and more functionalities have been included, resulting, in several domains, in a dramatic shift in system complexity. To face this problem, monolithic system designs have been progressively replaced by modular approaches, in which the overall system is no longer considered as a single entity in the design process, but rather the composition and coordination of a set of reusable components [23]. In addition, in the current era of ubiquitous environments with the serendipity of resources and the *mobility* of devices, systems have to operate in an open world by enabling the integration of existing cyber-physical systems and by achieving goals that are not achievable separately. In this context, it is often necessary to provide *critical* functionalities by strictly guarantying non-functional requirements as performance, timeliness, dependability and safety by combining several design and evaluation techniques.

A growing interest has moved to *mobile safety-critical applications* which have the potential of causing detrimental consequences in case of failure such as loss of physical property, physical harm, and loss-of human lives. For these reasons they require the utmost care in their specification, design, implementation, verification & validation, operation and maintenance to assure that dependability properties are actually achieved. An open world with the presence of mobile devices makes the provision of such highly resilient and available services difficult. Mobile nodes may communicate through wireless networks, possibly using general purpose operating systems and COTS-based devices, thus leading to uncertainty on the reliability of the operations in place. Cooperative collision avoidance in vehicular networks [52] and systems to support teams of workers operating in dangerous, dynamic and non-controlled environments [53] are two prominent examples of such mobile safety-critical applications. These new scenarios pose several threats in terms of dependability, timeliness and security, due to the challenging operating conditions caused by mobility, wireless connectivity, and the use of off-the-shelf components.

Recently this vision brought by mobile safety-critical applications has been emphasized by the *System of Systems* (SoS) concept [1], [3], [24], which has emerged in many fields of applications. In the SoS paradigm, a complex system or application is the result of the integration of many independent, autonomous Constituent Systems (CS), which are brought together in order to satisfy a global goal under certain rules of engagement. Differently from traditional monolithic systems and COTS modular approaches, SoSs better support criticalities in complex systems since they enable reasoning upon the complex overall SoS behavior, which results from complex interactions and dependencies of

Submitted on 10 December 2015 to IEEE Systems Journal.

This work has been partially supported by the European research project FP7-2012-324334-AMADEOS. The projects ALARP [17] and HIDENETS [19] were supported by the European Community while Secure! [33] was supported by Tuscany Region.

Andrea Bondavalli, Andrea Ceccarelli, Paolo Lollini, Leonardo Montecchi, Marco Mori are with the Department of Mathematics and Informatics of the University of Florence, Viale Morgagni 67/A, 50134, Florence, Italy (e-mail: bondavalli@unifi.it, andrea.ceccarelli@unifi.it, paolo.lollini@unifi.it, leonardo.montecchi@unifi.it, ma.mori@unifi.it), and with the Consorzio Interuniversitario per l'Informatica (CINI), Florence, Italy.

the underlying heterogeneous and autonomous CS.

Our contribution consists in showing how an SoS vision can ease designing, developing and maintaining mobile safety-critical applications through the identification of their major challenges and the corresponding application of viable solutions. To this end, we carried out a retrospective analysis of three mobile safety-critical applications in light of SoS design principles. By considering such applications as SoS instances, we made it possible to:

- exploit solutions already experimented for mobile safety-critical applications domain in the context of SoSs, thus broadening their applicability;
- systematically characterize and structure the challenges of mobile safety-critical applications.

The mobile safety-critical applications considered in this paper come from the authors' past experience on three different research projects for which a set of solutions have been proposed and experimented in specific contexts. Through our retrospective analysis, we select and tune among the former solutions the ones better suited to solve the challenges of mobile safety-critical SoSs and we highlight the benefits of their applications to the three SoS scenarios.

The paper is organized as follows. In Section II we discuss the three different motivating scenarios. We present in Section III the main challenges of mobile safety-critical SoSs and we consequently discuss in Section IV related work partially covering the former identified challenges. From Section V to Section X we separately discuss the identified challenges along with a set of viable solutions. Finally, Section XI concludes the paper with a summary of achievements and possible future research directions.

A. SoS Basics

This subsection provides insight into SoS basics which will be necessary to instantiate mobile safety-critical applications as SoS instances. Different definitions of SoS have been proposed in the literature according to different real-world applications in different areas, among others crisis management and vehicular networks. The one we adopt is the following [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.” Constituent Systems (CS) are either existing legacy systems, possibly belonging to different organizations, or newly developed components either hardware or software. CSs may also consist of controlled physical objects and humans interacting to provide a given service through sensors. The integration of CSs in an SoS is achieved through appropriate communication facilities.

SoSs may have a different degree of control and coordination according to the widely accepted classification proposed in [5], which identifies four different categories, namely directed, acknowledged, collaborative and virtual. A *directed* SoS is managed by a central authority providing a clear objective to which each CS is subordinate; the CSs that form the SoS may operate independently, but they are subordinated to the central purpose. An *acknowledged* SoS has a clear objective but the CSs might be under their own control thus funding an authority in parallel with the SoS. In a *collaborative* SoS, the central management organization does not have coercive power and 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. The degree of control and coordinated management of the CSs that form the SoS is relatively tight in a directed SoS, but it gets looser as we move to the acknowledged, collaborative and finally virtual category.

II. Motivating Scenarios

We present three motivating scenarios posing the challenges relevant to mobile safety-critical SoSs from the authors' experience on three different research projects.

ALARP (Railway Automatic Track Warning System Based on Distributed Personal Mobile Terminals) [17] consists in studying, designing and developing innovative Automatic Track Warning Systems (**ATWS**) to improve the safety of railway trackside workers. Its main objective is to notify working gangs along train lines, alerting them of trains and other rolling stocks approaching the worksite [18].

The considered ATWS scenario exploits a network of wearable mobile devices, by means of which the system has to reliably dispatch risk events to the workers. Localization facilities are required to classify such risk events either as an alert, if the worker is in a dangerous area (called red zone), or as a warning if the worker is located in a non-dangerous area (called green zone). The red zone comprises the track on which the train is approaching, and the adjacent area within a minimum safety distance (established by national railway regulations). By means of specific health sensors (e.g., heart rate sensor) the system should also be capable of monitoring the health of workers, and map their positions to identify the workers at risk, i.e., those not responding due to health problems, or located close to the track while a train is approaching. According to railway regulations, a risk event should be delivered with sufficient time in advance to allow the workers to reach a green zone.

To be successfully employed in a railway worksite, ALARP ATWS must satisfy the following non-functional requirements: i) designed to be self-powered, so that it does not require external power supplies; ii) composed of portable and wearable devices communicating by wireless links; iii) adopt a user-friendly and trusted interface for workers; iv) able to operate in different working places such as open line, stations, tunnels, bridges; v) able to operate in different working conditions (e.g., night or daytime) and weather conditions (e.g., fog, snow, rain, high/low temperature); vi) composed of low-cost equipment and rely as much as possible on off-the-shelf (OTS) components to achieve low production costs. As track-side workers are exposed to severe risk of harm, ATWS has strict requirements on safety: it is required to satisfy at least the Safety Integrity Level 2 (SIL 2) according to railway standards. The latter propose classes for the safety of equipments with associated qualitative and quantitative requirements. Quantitatively, SIL 2 means that the Tolerable

Hazard Rate per hour is required to be $10^{-7} \leq THR < 10^{-6}$ [44].

The **Secure! crisis management system** [33] provides the support for the public and private security management by exploiting a combination of social media and crowd sourcing/sensing technologies. Secure! is a novel Decision Support System (DSS) for emergency management. It exploits information retrieved from different types of sensors to detect critical situations and perform the corresponding reactions. Sensors range from web spiders surfing social networks, humans collaborating via a dedicated Secure! application installed on their mobile devices and sensors networks which are installed in an area of interest. Secure! should also be able to detect critical situations before they happen analyzing micro-events provided by the social media and correlating them with historical data and the micro-events from other sources. For example, threats to people or things may be detected making a syntactic analysis of the text content extracted from twitter: searching for particular keywords and then recognizing the intentions of a spiteful person.

One reference scenario for the Secure! Framework is a "world-heritage protection" application supporting the surveillance and protection of monuments and people against public demonstrations, acts of vandalism, armed robbery and so on. Tweets, phone calls, images, videos related to mobile devices in the area, if appropriately jointly interpreted, can be good predictors of critical situations. On the other hand, mobile devices and surveillance tools (possibly present on-the-field) are the basis for deploying reaction strategies, e.g., by supporting alarm notifications (early warning messages).

This scenario poses different security and resilience issues. Concerning security, the target application should acknowledge only trusted information in order to avoid the detection of wrong scenarios. At the same time, the target application should reliably collect trustworthy data as the base for creating appropriate reaction strategies.

HIDENETS Vehicular Network [19] focused on the development and analysis of end-to-end resilient solutions for distributed and mobile applications in ubiquitous communication scenarios, assuming highly dynamic and unreliable communication infrastructure [20]. One of the reference scenarios was a collisions avoidance system where vehicles cooperate to prevent chain collisions, or to reduce their severity (especially in high-speed roadways). This required that emergency information is propagated among vehicles much quicker than in a traditional chain of drivers reacting to brake lights of vehicles immediately ahead. One amenable solution at the time of the project was the adoption of car platooning techniques, in which vehicles operate safely as a platoon on a highway, controlled by the head vehicle.

A relevant safety critical requirement for the platooning is that cars have to maintain a minimum distance to their front car, taking into account that the distance will vary over time according to the car speed and the environmental conditions. For this purpose, fault tolerance mechanisms have been introduced both at the middleware and communication layer.

III. Challenges for mobile safety-critical SoS

Different challenges are posed to design, develop and maintain mobile safety-critical SoSs. In this section we present these challenges as they emerge from the motivating scenarios (see Table I). By means of the scenarios we will make evident the validity of the resulting challenges and we will propose corresponding solutions. Noteworthy, we do not aim at creating new solutions for SoSs but we do collect in a single framework the challenges in designing mobile safety-critical SoS along with solutions as they have already been presented for traditional distributed systems. Our identified challenges represent different viewpoints, i.e., angles of analysis for a mobile safety-critical SoS, namely *architecture*, *dynamicity and evolution, emergence, governance and constraints*, *handling of time*, *dependability and security* (see Table II).

Challenges concerning *architecture* are related to *multi-criticality* and *hierarchical design and control. Multi-criticality* allows safety-critical functionalities to be managed differently from non safety-critical functionalities; *hierarchical design and control* is meant to support the collaboration/cooperation among highly distributed CSs. In the following sections, the architectures of Secure!, ALARP and HIDENETS are introduced, resulting in three different reference architectures specifically identified to match different challenges. In particular, multi-criticality is discussed with reference to the HIDENETS architectural solutions, while Secure!, ALARP and HIDENETS offer three different examples for the hierarchical and holarchical [34] architectural design.

Dynamicity supports the achievement of the overall objectives despite external changing condition and failures while *evolution* refers to long-term variations of the system to support changing business requirements thus improving system agility [55], [56]. Examples of dynamicity and evolution and related approaches to deal with them are reported for HIDENETS and Secure!, while ALARP is left aside in this viewpoint. It is also worth noting that HIDENETS included specific answers to the evolution of the system, bringing design approaches that promote and are compatible to long term evolution.

By governing *Emergence* it is possible to avoid detrimental situations resulting from (cyber or physical) interactions among CSs and to achieve solutions that are not achievable by only considering CSs in isolation. Examples of emergence in Secure! and HIDENETS are reported, and preliminary solutions to capture emergence phenomena are envisioned.

Table I - Viewpoint-driven analysis of the motivating scenarios

	Multi-criticality	*Hierarchical design & control*	*Dynamicity & Evolution*	*Emergence*	*Governance & constraints*	*Handling of time*	*Dependability & Security*
ALARP [17][18]	✓	✓	×	×	✓	✓	✓
Secure! [33]	×	✓	✓	✓	✓	×	✓
HIDENETS [19][20]	✓	✓	✓	✓	✓	✓	✓

Governance and constraints support the enforcement of rules and procedures to be followed in order to achieve the overall objective appropriately. HIDENETS, Secure! and ALARP are regulated by different governance models, that span from regulations to societal aspects, and different sets of constraints as privacy limitations or available resources.

Handling of time supports the achievement of a global time basis as it is necessary in highly-distributed safety-critical environments. In fact, HIDENETS and ALARP required the design and development of specific solutions to achieve accurate time-keeping and real-time communication in mobile and possibly hostile environments.

Dependability and *security* support assurances of safety-critical functionalities by operating at three different levels, i.e., tackling the *environment* uncertainty (discussed for ALARP, HIDENETS and Secure!), *monitoring* the current situation (monitoring solutions are discussed for ALARP and Secure!, including position monitoring) and performing the consequent *assessment* (in particular, model-based assessment approaches in ALARP and HIDENETS are reported).

The discussed challenges, even if with different names, have already been considered in the literature of SoS [1], [2] but not much effort has been devoted to the mobile dimension. Some of them (interoperability, emergence, handling of time, dependability, security, dynamicity, evolution, governance and constraints) have been addressed without a particular focus on mobility and the consequent exacerbated threats to the achievement of dependability requirements. Others (multi-criticality and hierarchical design and control) have been already addressed for traditional distributed systems, but still have not been thought in the context of mobile safety-critical SoSs. Nevertheless, solutions are applicable in the SoS context as we prove with the support of our motivating scenarios. It is worth noting that the above challenges show cross-cutting aspects as it will emerge in the next sections.

IV. Related Works

We provide an analysis of the most representative approaches capturing the challenges identified in supporting design, development and maintenance of mobile safety-critical SoSs. To the best of our knowledge most of the approaches take into account just a few challenges while only a few provides a broader coverage of the challenges. The latter have been considered at different extent and for different application domains.

The work in [3] introduces five viewpoints which have to be carefully considered in an SoS. *Operational independence* consists in the capability of disassembling an SoS while still keeping the resulting CSs able to operate independently. *Geographical distribution* allows an SoS to achieve its goal given that CSs are distributed spatially and communication facilities have to be provided. *Emergent* behavior enables to express the SoS purpose through the collective actions of the system participants. Finally, e*volutionary/adaptive development* supports short-term and long-term CSs reconfigurations. [4] is another attempt in considering different SoS viewpoints jointly in a single framework, namely *interdisciplinary*, *heterogeneity* of CSs and *networks of systems*. Finally the approach presented in [7] introduced *autonomy*, *connectivity*, *belonging*, *diversity*, and *emergence* as key viewpoints. The approaches in [3], [4] and [7] consider only a subset of challenges raised by mobile safety-critical SoS with little or no support of current practice technologies.

Table II - Viewpoint-driven table of content

Viewpoint	*Section*
Architecture	Section V
Multi-criticality	Section V.A
Hierarchical design and control	Section V.B
Dynamicity & Evolution	Section VI
Emergence	Section VII
Governance & constraints	Section VIII
Handling of time	Section IX
Dependability & security	Section X
Environment	Section X.A
Monitoring	Section X.B
Assessment	Section X.C

A set of funded research projects are supporting particular phases of an SoS from different perspectives. The DYMASOS project [8] aims at developing new methods for the distributed management of large physically connected systems having a distributed autonomous management and global coordination. Its main concerns are related to *governance and constraints*, *hierarchical design and control* and *dynamicity*. The Local4Global project [9] aims to develop, test and evaluate a generic integrated and fully-functional methodology for controlling SoSs where CSs react and interact depending only on their local environment. The project provides means to support optimization of global qualities (i.e., constraints) and it considers as main focus concerns like *hierarchical design and control*, *dynamicity*, *evolution*, *emergence* and *handling of time*. The COMPASS project [10] aims at integrating engineering notations, methods and tools in the modeling and analysis process of SoSs. Architectural concerns play a key role as well as the provision of *fault-handling*, *responsiveness* and *emergence* handing with respect to overall objectives (i.e, constraints). The DANSE project [11] aims at bringing evolution and adaptation within the SoS life. Architectural concerns along with *dynamicity*, *evolution* and *emergence* are among the key assets for DANSE project. It also provides supports to the achievement of *dependability and security* requirements and it guarantees the *timely* response of an SoS. The AMADEOS project [12] aims at defining an architecture for SoS management, including the means for monitoring the system itself and the environment, predicting possible future behavior and the support for reacting to *adpapt*/*evolve* the current functional and non-functional requirements. Starting from the identification of the key requirements for SoSs [46] and from the definition of an SoS conceptual model, the AMADEOS architectural framework allows a *hierarchical* SoS design and it supports the achievement of *time-dependent*, *emergent* and *multi-critical* requirements of an SoS.

From the analysis of the literature we can claim that almost all the above approaches and projects provide means to achieve dependability and security but with little or no attention to the mobility dimension and environment uncertainty. Handling of time has been considered but with no support in providing a trustable global time base among CSs and challenges like multi-criticality have no valuable support for its enactment on SoS environments.

V. Architecture

The first critical aspect in an SoS is its architecture defined in terms of heterogeneous CSs, interacting each other through cyber or physical channels; thus a specification of interfaces among CSs has to be properly defined. An emerging interest towards Relied Upon Message Interfaces (RUMIs, [37]) and Relied Upon Physical Interfaces (RUPIs, [38]) among CSs has recently been raised to establish the boundaries between two interacting CSs. RUMIs establish the data that are exchanged and the exact timing of message exchange, while RUPIs enable the physical exchange of things or energy among CSs.

For the ALARP scenario, we consider an SoS architecture (Figure 1) involving the following main CSs: i) the track-side train presence alert device (TPAD), able to sense an approaching train on the track, ii) a set of wearable wireless Mobile Terminals (MTs), to inform the workers about possible approaching trains and other events that could put at risk their safety. CSs collaborate through a wireless communication infrastructure. This represents an instance of a directed SoS where CSs are subordinated to a centrally managed purpose, i.e., improving the safety of railway trackside workers.

For the Secure! scenario we consider an architecture with different CSs (see Figure 2) aiming at receiving, collecting, homogenizing, correlating and aggregating raw events, in order to detect emergency scenarios. A set of CSs is devoted to collect raw events (e.g., tweet, images and videos) from any of the following sources: (i) social media and web sites, i.e. social networks, (ii) mobile devices and their embedded sensors (GPS, gyroscope, accelerometer, thermometer, proximity sensors), including human sensors, (iii) sensor networks in critical infrastructures. Source data collection and integration CSs are responsible for filtering and collecting raw events, e.g., video surveillance camera producing videos in a certain area and mobile devices additionally producing tweets and images. Higher level CSs combine low level events into a higher level of understanding to define situational pictures, and finally Secure! app and Service CSs (DSS level) deliver alert notifications to citizens. As for the ALARP SoS, also the Secure! SoS is directed since its CSs are subordinated to a centrally controlled purpose, consisting in the management of public and private security.

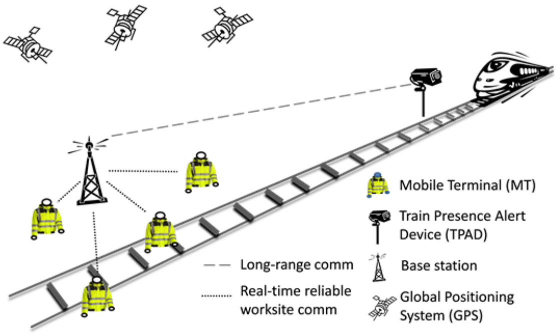
Mobile Terminal (MT)
Train Presence Alert Device (TPAD)
Base station
Long-range comm
Real-time reliable worksite comm
Global Positioning System (GPS)

Figure 1 - ALARP architecture

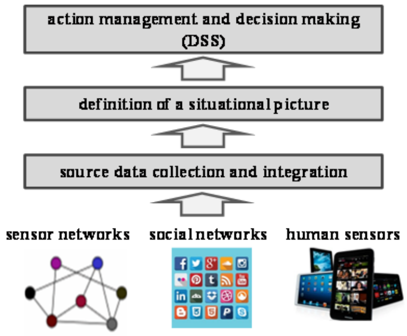
action management and decision making (DSS)
definition of a situational picture
source data collection and integration
sensor networks
social networks
human sensors

Figure 2. Secure! architecture

For the HIDENETS scenario, we consider an architecture where CSs are cars that interact to exchange all required information to avoid collision (see Figure 3). Each CS keeps a minimum distance to its preceding CS to avoid collision and a maximum distance to its rear car to avoid fragmentation of the platoon. The car driver interacts with the car through a set of available actuators, i.e., the accelerator, the brake and the platoon activation button. The latter, when activated, forces the car to take part to a platoon of cars. Each car, in turn, entails a set of different CSs; among others it is equipped with a sensor to determine its position (GPS receiver), a sensor for evaluating distances to preceding and rear cars (distance sensors) and a sensor to determine its speed (speed sensor). This information is collected and exchanged among cars (through wireless network) in order to regulate cars speed, to form a platoon and finally to avoid collisions. Besides the car-to-car communication through ad-hoc wireless networks, cars can also communicate with a fixed infrastructure, either as an alternative network route to reach other vehicles, or to use other services (e.g., entertainment). The HIDENETS SoS represents an instance of collaborative SoSs in which CSs voluntarily collaborate to fulfill the shared purpose: avoid collisions. This purpose is not centrally imposed to CSs, which may also de-activate the platoon activation button thus behaving independently one another.

Taking the perspective of the SoS architecture for mobile-safety critical applications, we have identified two relevant challenges to be solved at the architectural level: i) multi-criticality, and ii) hierarchical design and control.

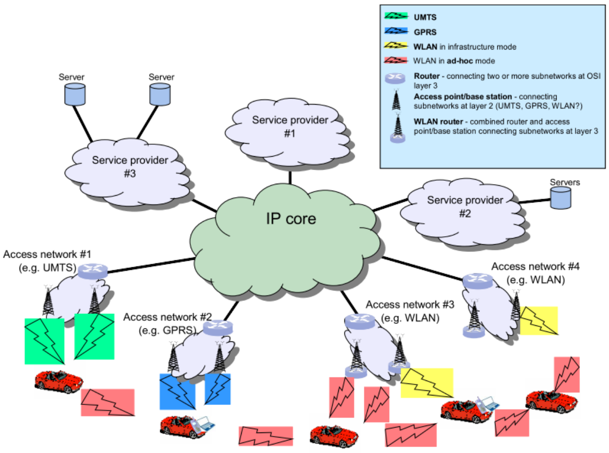
UMTS
GPRS
WLAN in infrastructure mode
WLAN in ad-hoc mode
Router - connecting two or more subnetworks at OSI layer 3
Access point/base station - connecting subnetworks at layer 2 (UMTS, GPRS, WLAN?)
WLAN router - combined router and access point/base station connecting subnetworks at layer 3
Server
Server
Service provider #1
Service provider #3
Servers
Service provider #2
IP core
Access network #1 (e.g. UMTS)
Access network #2 (e.g. GPRS)
Access network #3 (e.g. WLAN)
Access network #4 (e.g. WLAN)

Figure 3. HIDENETS architecture

A. Multi-Criticality Requirement

Architectures of mobile safety-critical applications shall be built considering that while some parts of the system may have strong safety-critical requirements, other parts may be not so critical. This raises an architectural concern on how to support these different system parts. To this end, we propose the adoption of *architectural hybridization* [29] which allows constructing SoSs made of two different parts: the wormhole on which simple but critical functionalities reside, and the payload on which non-critical functionalities are placed.

In the HIDENETS scenario the wormhole contains an authentication service that, using a public-key infrastructure model, collects sensible information (speed and position) from nearby cars after they have been authenticated. A reliable and self-aware clock [39] is also placed in the wormhole in order to obtain an assured maximum time deviation from an external global time reference. This is essential to timestamp positioning data shared among cars, thus allowing a safe localization of close cars. On the contrary, the payload contains functionalities like an intrusion tolerant agreement service [48], which is adopted to establish a consensus on the platoon speed, and other QoS communication capabilities exploited to tune a decision control algorithm. These functionalities, although important for the car behavior, do not show criticalities to be protected in a wormhole. A quantitative evaluation of end-to-end dependability and quality of service metrics of complex HIDENETS scenarios has been presented in [20], whose results have been used for the successful apportionment of the different dependability requirements to the different parts of the system.

B. Hierarchical Design and Control

The control of the SoS architecture shall be achieved hierarchically by considering the two-level perspectives identified by the SoS and CS layers. Depending on the adopted level of detail, a CS can in turn be considered an SoS on its own. For example, in ALARP, the MTs and TPADs are CSs of the overall system. However, each of them can also be considered an SoS, in which its heterogeneous components are the CSs. This is much more evident in the HIDENETS system, where each vehicle is a CS with respect to the HIDENETS infrastructure, but also an SoS with respect to the different electronic and mechanical components which constitute it.

Depending on the adopted perspective, an SoS can form a *formal hierarchy*, or a *non-formal hierarchy (holarchy)* [35]. In a formal hierarchy each subsystem at level *n* is linked vertically by a reporting and control relation to its controlling system at Macro-level *n-1* (i.e., the level above) [36]. For example, parts of the Secure! SoS present this kind of interactions: the Secure! SoS includes, amongst other, the human sensors, which are CSs that constitute a certain macro-level and do not interact each other. In a holarchy there are horizontal interactions among the related subsystems at Macro-level *n* that lead to the formation of a whole with its own specific properties from the point of view of the level above (Macro-level *n-1*). Starting from the top, it is possible to identify at the Macro-level, i.e., the whole, the purposes of the SoS (e.g., HIDENETS Vehicular Networks and ALARP ATWS). At the Macro-level-1, the CSs (e.g., HIDENETS cars) interact with each other in order to realize the purposes at the Macro-level. At the Macrolevel-2, the CSs are decomposed in subparts such as individual parts within a car. In principle, this iterative decomposition can always be carried to a further level.

Clearly understanding a *multi-layered hierarchical* SoS structure, eliciting the relations between components at a certain level and between components at the upper and lower levels, is a prerequisite for successfully carrying out activities like documentation, modeling, evaluation. Layers might be modeled according to different notations depending on the details of interest and the involved stakeholders. To this end, semi-formal languages like UML, AADL, SysML, and Architecture Description Languages (ADLs, [49]) in general, form a valuable support to the layered modeling of SoS architecture and interactions among its CSs. Furthermore, UML profiles addressing specific non-functional properties exist (e.g., MARTE for real-time concerns [50], SoS extensions to SysML [51]), which can be exploited to document SoS non-functional properties in details.

VI. Dynamicity and Evolution

Dynamicity and evolution are two important challenges of SoS in general and they play a key role in mobile safety-critical applications. Dynamicity refers to short-term SoS changes to be taken in response to environmental variations or components failures, aiming to achieve a certain goal. Evolution refers to long-term changes that are required to accomplish variation to the requirements in face of an every/changing environment.

Evolution. A relevant challenge consists in supporting the SoS in evolving itself to follow requirements variations. This aims to support the growing of intra-domain interactions and communications, to ease the adoption of new technologies, to support fragmentation of the operators and to support the adoption of new standards. Solutions to these problems are basically architectural and they consist in adopting a design for evolution approach, which consists in having flexible and robust architectures easily adaptable to changes. This can be achieved by adopting (i) *multi-layered hierarchical* approaches and (ii) consequent applicable generative *Model-Driven Engineering* (MDE, [22]) *techniques* thus promoting reuse and decoupling among CSs. We envision the adoption of a multi-layered system specification, which should be easily transformed to (or derived from) each other when required, and be kept consistent in order to react to evolution. Target specification should also serve as input to development steps like code generation, formal verification and testing.

In the context of HIDENETS we adopted MDE techniques for design and evaluation purposes. A key feature of MDE is model transformation [30], which is used to refine models, apply design patterns, and project design models to various mathematical analysis domains in a precise and automated way. Challenges posed by system evolution are partially alleviated by adopting a MDE approach: in this perspective, coping with evolution involves modifying only a subset of the system model or documentation, while all the other artifacts (e.g., code, analysis models, domain-specific views) can be re-generated automatically by model-transformation techniques. Furthermore, model transformations have also been widely used for the analysis of non-functional system properties, e.g., dependability, schedulability, performance. Two main reasons typically drive the application of such techniques: i) complex mathematical models can be automatically generated from the system design, without the intervention of specialized figures; and ii) the resulting model is by construction consistent with the architectural design, i.e., human mistakes are avoided. As shown MDE approaches can be effectively used to model and quantitatively assess key dependability properties of complex, evolving systems. As shown in [20], for example, complex and evolvable HIDENETS scenarios can be modeled and analyzed through the definition and implementation of a transformation workflow based on the automatic composition of sub-models into higher level models, thus allowing the quantitative assessment of end-to-end dependability properties at different systems levels.

As an example, an evolution enacted in the HIDENETS scenario consists in augmenting the car with remote control facilities. To this end, the communication layer among cars (CSs) has been improved with a possible new CS managing the new type of interactions to enact remote decisions through car actuators (accelerator and brake). Following the approach defined in [20], the impact of adding the new CS on the system dependability properties can be automatically assessed deriving a new quantitative analysis model from the augmented system model.

Dynamicity. Managing dynamicity prevents the SoS in deviating from the achievement of its goal despite changing environmental conditions and failures To this end, in the HIDENETS scenario we have adopted algorithms able to adjust their behavior according to external factors. Different conditions threat the collision avoidance goal, e.g., the number of connected cars (CSs) can quickly change, new cars can be identified, communication with one or more fixed infrastructure devices may fail. The solution proposed for this scenario consists in providing cars with a decision control algorithm made of two different variants, which is exploited to alternate the control behavior with the enactment of the evaluated decisions. Decisions consist in determining the target car speed to achieve, which depends on its current position and speed and on the positions and speeds of its neighbor cars. Once a decision is taken the control part of the algorithm is re-iterated only after the time for enacting the decision has expired. This scenario showed that it is possible to achieve a fixed goal (i.e., collision avoidance) by adopting *context-aware adaptive algorithms*, which dynamically adapt the SoS according to internal and external conditions.

Adaptivity, beyond software, may also involve reconfigurations to the SoS architecture. In the Secure! crisis management system scenario, different stakeholders may dynamically establish connections with the aim of managing and solving critical situations. In the literature approaches have been defined to design *adaptive architecture* enabling the automatic reconfiguration of components [38].

It is worth noting that dynamicity and evolution scenarios pose threats to the achievement of dependability and security requirements, e.g., it is not sufficient to adopt a good level of reuse and decoupling to maintain the SoS safe after its evolution. We refer to Section X for a broader discussion on dependability and security.

VII. Emergence

The concept of emergence has been defined and extensively studied in biology, philosophy and artificial life. An emergent phenomenon can be defined as a phenomenon that manifests itself at the macro-level but it is not observable at the micro-level. This phenomenon is not a-priori positive or negative but this depends on the observer's criteria [36]. Emergence is a property of SoSs, and it can be expected or unexpected, detrimental or non-detrimental (positive). In many situations, the first appearance of the emergent properties is unforeseen while afterwards it makes no surprise anymore and can even be explained through laws of science. In systems that must adhere to strict non-functional properties, e.g., safety-critical and mission-critical systems, emergent phenomena may be an unexpected source of failures, possibly leading to catastrophic failures of the system functionalities.

The rationale beyond emergence is that the SoS is not simply the sum of its CSs. Indeed emergence is strictly related to the level of knowledge (or to the level of ignorance) on a specific system. Since computers are deterministic machines, the overall SoSs behavior can be seen as a function of CSs behavior (sub-functions) composing it. Ideally, having the full knowledge of individual components' behavior and of their interactions would allow the overall system's behavior to be fully predicted. In mobile systems, however, such kind of knowledge is unrealistic given the unreliability of communication links and the dynamicity of mobile devices. This aspect is even worse in SoSs, where individual CSs are possibly owned by different stakeholders, which may limit the information publicly available on the subsystems [37].

Typical examples of detrimental (negative) emergence in the vehicular domain are traffic jams. Such events stem from a combination of multiple factors, e.g., individual decisions, habits. More specifically, in the HIDENETS SoS, traffic jams may cause a saturation of the wireless medium, which, in turn, may prevent the activation of car platooning techniques (e.g., due to high latency in message delivery).

An example of positive emergence is offered by the Secure! crisis management systems, where different entities (intervention teams, common people using the Secure! App), which commonly operate individually, are able to quickly interact thanks to the activities of Desktop operators. This leads to an enhanced ability to read the ongoing scenarios, define intervention strategies, and coordinate on-field operations. Detrimental emergence examples can be identified in the unplanned, excessive number of human sensors in Secure!, generating a multitude of events that the elaboration process is unable to appropriately correlate for providing exhaustive information to the Desktop operator.

As per today, emergence is a relatively new field of research and no widespread, widely accepted, approaches to deal with emergence in SoSs are present in the literature. Promising appearing solutions are *simulation-based* and *Goal-Oriented Requirements Engineering* (GORE) *approaches.* The former have been defined in the context of DANSE methodology [11] to discover the existence of emergent phenomena by means of applying simulations to the models representing the SoS and its interacting CSs. The latter approaches apply the concept of emergence to the Requirements Engineering (RE) stage [31] and match it to the ignorance on the system. As presented in [32] the designer can perform a preliminary assessment of the level of ignorance associated to parts of the system by means of an iterative procedure included in the goal-oriented requirement engineering process.

VIII. Governance and constraints

Governance, including the adherence to standards and procedures, can significantly hamper the realization of safety-critical applications on an SoS architecture. Distributed ownership of individual components is a challenge for any SoS [27]. Governance becomes significantly more complicated and must change to accommodate the business requirements of an SoS. Additionally, new components and applications must adhere to pre-existing regulations and procedures, although they could be in contrast with the purpose of the application itself.

Considering our reference projects, governance issues in the ALARP ATWS arise from existing regulations in the railway domain, which should be applied to several components of the system including the mobile terminals and the wireless communication. Although at EU-level safety standards are shared, rules and procedure for operating in a worksite vary from country to country, thus requiring specific configuration and usage procedures of the ALARP system. Additional constraints traditionally introduced by railway stakeholders include costs, i.e., the final solution should have a competitive cost, and power efficiency, i.e., all the devices constituting the system, including TPADs, should be able to stay continuously powered on for the entire working day.

Vehicular ad hoc networks as foreseen in HIDENETS provide a prominent example of regulations gaps with respect to autonomous driving applications. Currently, there are ongoing discussions about liability in the event of an accident involving a vehicle driven through an interaction between in-board and off-board components; furthermore, the insurance industry will need to adapt their current risk models [28].

As per today, similarly to emergence, no approaches for designing systems focusing on governance or SoS constraints are present. From our experience in HIDENETS, ALARP and Secure!, the typical approach is to collect and adhere to the available standards and legislations, and to the available constraints as technology, assets, financial resources, expected lifespan and system life. Governance and system constraints are expressed in terms of requirements that are set at the beginning of the design phase, and that may span through several areas, ranging from dependability and security assessment to trust and privacy assurance, potentially including societal aspects. For example, ALARP strictly follows the railway standards; Secure! includes national legislations for the management and disclosure of privacy-sensitive information; in HIDENETS, despite the involved mobile vehicular networks can potentially move through different countries, there are no cross-borders legislations on their usage. Therefore, specific techniques for *flexibly complying with regulations of different nature* should be defined and adopted to cope with cross-liability issues among multiple stakeholders.

IX. Handling of Time

The issue of time awareness in SoSs is receiving growing attention by the industrial community. In large cyber-physical SoSs the availability of a global sparse time appears more and more mandatory to reduce the complexity of understanding, designing and implementing SoS [25]; however, CSs typically rely on their own, unsynchronized, clocks. Techniques are therefore needed to provide CSs with a global view of time.

ALARP and HIDENETS are required to dispatch events within the expected time bounds. The environment, the mobility of devices and the dynamicity of the system in general are major threats in achieving predictability. For example, interferences due to obstacles can lead to significant unpredictability in message transmission delay; these delays in the ALARP scenarios were quantified in [15], [21].

Specific solutions for *real-time communication* and *clock synchronization* are then required. Amongst the many that have been developed through years, in ALARP the communication network at the worksite is based on a centralized approach, where all MTs use a two-hop communication via a coordinator, also called base station. The communication layer offers reliable broadcast and reliable unicast transmissions, which are both implemented by the Timed Reliable Communication (TRC, [15]) protocol, designed on the basis of the protocol presented in [16], and adapted to ALARP requirements and communication scenarios. This synchronous protocol, based on Time Division Multiple Access (TDMA), relies on a time-slotting approach for polling MT nodes by the coordinator and it is implemented on 802.11b/g/n. Primarily targeted at disseminating safety-critical events, the protocol does not require (and therefore does not offer) agreement and validity properties. This also permits time savings in terms of a shorter overall worst case message delivery delay. The TRC was assessed quantitatively in [15], [21].

The clock synchronization approach adopted in ALARP and HIDENETS is based on NTP over GPS, and NTP over Internet. Given the possible unreliability of such transmission channels, the synchronization mechanism is complemented by a resilient clock for self-aware communication. Providing both the current time and the synchronization uncertainty, the HIDENETS clock is capable of monitoring synchronization quality and detecting clock failures or poor synchronization. The resilient clock has been assessed experimentally in [39].

X. Achieving Dependability and Security

In this section we discuss the main threats to dependability and security that in our opinion are particularly critical when facing an SoS approach for building mobile systems.

- **Environment.** As SoSs are fundamentally composable systems, with a high degree of uncertainty on their boundaries, the environment may unpredictably change, or it may be so vast that it is difficult to describe. Thus (mobile) SoSs shall include solutions to deal with possibly different environment and operating conditions, or being able to adapt to them.

- **Monitoring SoSs.** Monitoring is a fundamental mean to observe if dependability and security properties of a system are satisfied [40]. Monitoring a mobile SoS means to devise adaptive monitors that are able to cope with different environments and variable number of interacting CSs, and adapt to emergence phenomena.

- **Assessment.** Assessment of mobile SoSs includes the challenges of assessing mobile, heterogeneous systems, with a high degree of evolvability and subject to emergent phenomena (possibly unexpected, thus not planned at design time and not tested for in lab).

A. Environment

Threats to resilience, safety, security, trust and privacy are often exacerbated by potentially adverse environmental conditions, on which the system has limited control. ALARP, HIDENETS, and Secure! are three examples of systems with mutable environments, i.e., they may need to operate in different environmental conditions. In fact, railway workers, vehicles, as well as rescue teams, may operate in bad weather conditions, close to places with electronic interferences, surrounded by obstacles which may reduce communication.

Fulfilling the existing non-functional requirements is particularly challenging: *i)* typically, device lifetime and communication are severely limited by scarcity of power; *ii)* the use of wireless links means susceptibility to link attacks; *iii)* mobile devices are susceptible to physical damage, and vulnerable to theft or subversion; *iv)* adverse weather affects communication, localization, and possibly battery efficiency.

Regarding point i), in ALARP power efficiency is required to guarantee that the device can operate for the whole working day. The finite state machine of the ALARP middleware includes a low power state, to specifically address issues on low power. The low power state is entered when the MT has low battery power; in this state the MT provides a subset of its functionalities in order to save batteries. In particular, during the *degraded operation mode*, the MT minimizes the number of messages the MT exchanges; from low power state, an MT can only move to a safe state or turn off.

Regarding point ii), we already discussed the communication solution in ALARP; the HIDENETS architecture includes the *Intrusion-tolerant Agreement* service [47], which provides nodes with various flavors of agreement protocols (e.g., binary consensus, vector consensus), allowing these entities to coordinate their actions. The protocols operate correctly provided that less than one third of the involved entities try to disrupt their operation.

In Secure!, the main focus was oriented to protect the devices from attacks, rather than from power outage or damage (the intervention team is supposed to be in group, with the possibility of easily replacing devices). Mechanisms are introduced to provide secure authentication, including biometric authentication for the intervention team and the Desktop operators, and trust mechanisms based on Secure Two Party Computation [54] technique. Regarding privacy, the access, exploitation and sharing of data in Secure! adopts an infrastructure of data policy where users define their access policies and the controller checks their compliance.

Regarding point iii), this is achieved through *rugged ad-hoc devices and HMI*. For example in the ALARP railway worksite, traditional HMI devices are likely to be ineffective (e.g., acoustic warnings may not be heard due to surrounding noise). Alternative solutions have been identified to provide reliable and safe notification of signals, as flashing lights (using diodes) installed on protective eyewear and ear-bone conductors to transmit acoustic signals via vibrations through the skull bone that are hearable even in noisy environments.

Regarding point iv), the ALARP middleware includes a *degraded state* to address issues on time uncertainty and localization uncertainty. MT activates specific procedures to mitigate such uncertainties, increasing the resources devoted to the execution of the algorithms for clock synchronization and localization [18]. In HIDENETS, the QoS Coverage Manager service provides support for applications to adapt to the available QoS. In essence, the idea to achieve the adequate level of *dependability and adaptation* is to ensure that a “coverage stability” property is satisfied, which means that the assumed bounds for fundamental variables (e.g., network delay) are secured with a known and constant probability. Detailed evaluation results of such dependable adaptation methodology have been presented in [41]. The evaluation was based on synthetic data flows generated from probabilistic distributions, as well as on real data traces collected in various internet-based environments; results show that it is possible to compute bounds in the order of 10% to 25% lower than the bounds produced by other conservative approaches, still securing the required coverage in all cases, and that the introduced processing overhead can be acceptable in many practical systems.

B. Resilient monitoring

While monitoring is evidently required in SoSs to timely detect errors, the mobility and loosely coupled interoperability of such mobile SoSs exacerbate some of the widely known monitoring challenges [42]. The monitor is expected to observe services resulting from emergent behavior of the SoS.

Considering our three guiding scenarios, the ALARP SoS despite being a distributed system includes only a limited number of nodes and running basically the same software. Thus, in the ALARP middleware it was possible to create a *distributed monitoring system*, which continuously verifies the operability status of the different MTs [17]. Thanks to the support of a real-time network and localization algorithm, the status and the position of the MTs is reported in real-time to the other MTs. References for the assessment of this monitoring system, comprising the real time network and the localization algorithm, can be found in [18].

Secure!, instead, offers a different scenario where a multitude of data sources is present (human sensors, information from the web and the social networks, information from infrastructure sensors) and in many cases such sources are just barely known by the system. Thus the Secure! SoS needs to use potentially unreliable information, coming from sources on which it has limited or no control, and with limited or no possibility to check the accuracy of the data received. Consequently, Secure! implements *trust mechanisms* aimed at “rating” the credibility of the different sources, and also event anomaly detection mechanisms to identify and discard falsified or biased data that could negatively impact the decision process. Evaluation using different data streams can be found in [60]. Furthermore, node mobility often introduces the need for some form of accurate *tracking* and/or *positioning solution*. While all the three projects we explored require positioning of people or vehicles, ALARP has severe requirements in terms of localization accuracy of workers. Low-cost GPS receivers are not sufficiently accurate to reliably identify if a worker is in a red or a green zone, as it was measured in [61]. A GPS-augmentation approach has been devised in ALARP, and it is presented and evaluated in [18]. In such approach, GPS data is combined with information provided by electronic fences (i.e. infrared links), which are placed in the worksite area at the border between red and green zones.

C. *Assessment of mobile dynamic evolutionary SoSs*

As the requirements and the system configurations evolve through time, it becomes extremely difficult to define a priori risks, failure modes, and RAMS [44] requirements before the deployment of the product. Models for resilience assessment cannot consider all possible evolutions of the system and its requirements; actually, most of the available methods are based on the construction and evaluation of models representing a static view of the system, with pre-defined requirements and system structure. On the other hand, performing measurements on the whole system is typically difficult or not even feasible, either because it is too expensive or dangerous, or because obtained results may be scarcely representative of the actual system operation, due the high variability of its properties and of the environment [47].

Mastering the evaluation of SoSs, maintaining the right level of detail, and at the same time accurately modeling all the interactions between system components require a holistic approach. Different evaluation techniques at different abstraction and decomposition levels are applied to solve sub-problems, and then combined, exploiting their interactions to support the system-level evaluation. Interaction among different evaluation techniques can occur by means of: i) cross validation; ii) solution feedback; iii) problem refinement [20].

An effective approach in facing dynamicity and evolution consists in combining *modeling and experimentation* (e.g., [11], [14]). We advocate the need of a dynamic model generation (and evaluation) process, capable to dynamically produce at run-time different models representing the current system state and conditions, and capable to feed the models' parameters with values coming from monitoring and experimental evaluation activities. In the evaluation of the ALARP system, model-based and experimental approaches have been combined at different abstraction levels. For example, in the evaluation of ALARP TRC protocol used for worksite communication [21], an experimental evaluation was first performed on a prototype implementation in a laboratory setup [15]. Similarly, in HIDENETS cross-fertilization among different methods was exploited, by feeding system analytical models with parameter values derived from simulations [20].

A key principle to realize such process is *modularization of models*: the system architecture is decomposed in "template submodels" [59], which can be replaced or refined as needed, provided that the input and output interfaces remain the same. The system model is then obtained by composing multiple instances of such templates, using different parameters settings. Adapting to changes in the system architecture is simply reduced to compose such instances differently, and/or modify the relevant parameters.

Such template-based approach has been applied in the evaluation of both the HIDENETS and ALARP systems, using the Stochastic Activity Networks (SAN) formalism. By coupling this approach with MDE techniques (see also Section VI), and with dynamic run-time monitoring [45], adaptive online evaluation can be achieved.

XI. Concluding Remarks

We described and proposed a set of challenges along with viable solutions to support design, development and maintenance of mobile safety-critical SoSs. Our challenges have been validated against three heterogeneous mobile safety-critical applications for which a set of current practice technologies have shown to be valuably applied. In Table III we summarize the proposed solutions adoptable for each of the identified challenges. Such proposed solutions are selected from the experience we made in the context of three large projects. It is evident that several other approaches are present in the state of the art and can be appropriately adopted. The objective of the possible dictionary of solutions presented here is to show that a plethora of approaches already exists that can be applied to architect SoS as well as traditional systems

Viewpoint	*Challenge*	*Possible Approaches*
Architecture and Semantic of Communication	Multi-criticality requirement	Architectural hybridization
	Hierarchical design and control	multi-layered hierarchical structure (formal hierarchy/holarchy)
Dynamicity	Achievement of a fixed SoS goal	Context-aware/Adaptive algorithms and architecture
Evolution	Achievement of a changed SoS goal	MDE techniques; multi-layered hierarchical structure (formal hierarchy/holarchy)
Emergence	Assessment of emergent phenomena	Goal-Oriented Requirement Engineering approach; Simulation-based approaches
Governance and Constraints	Enforcing governance and constraints	Flexibly complying to regulations of different nature
Handling of Time	Creating a global sparse time	Clock synchronization; Real-time communications
Dependability and Security		
Environment	Device lifetime	Degraded operating modes
	Faulty communication links	Intrusion-tolerant agreement solutions
	Poor device protection	Rugged ad-hoc device and HMI
	Adverse weather conditions	State degradation to mitigate uncertainty; Dependable adaption
Monitoring	Insufficient knowledge of the CSs governing the monitoring system; Introduce accurate localization	Distributed monitoring approaches; Trust mechanisms for unreliable data source; Tracking and positioning solutions
Assessment	Performing measurements on real instance of SoS; large state-space in modeling SoS	Modeling + Experimentation Modularization of analysis models

Table III - Challenge-driven dictionary of solutions for mobile safety-critical SoS

following our proposed viewpoint-based perspective.

From our analysis of the challenges, we can claim that mobile safety-critical SoSs require the reconsideration of the traditional approaches for system engineering. Such an SoS-oriented approach to system engineering demands a change in the traditional perspective for system design, assessment, implementation, deployment and evolution. Challenges traditionally under considered when building systems become here central (e.g., emergence, governance, or multi-criticality). Moreover, other challenges are intrinsic to the SoS-oriented 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. Looking back and reconsidering the three input research projects, we observed that a set of problems could have found different solutions using an SoS vision, thus taking advantage of the viewpoints defined in this paper. For example, the design process would have benefited by the identification of challenges at earlier design stages. Moreover it could have been carried out in a more systematic manner also exploiting the relations among viewpoints. In addition, by explicitly identifying RUMI and RUPI interfaces it would have been possible to explicitly design for emergence and characterize emergent behaviors resulting from the cyber and physical interactions of CSs. This would also have eased the identification of interoperability problems which may be difficult to capture without explicitly defining physical channels. Additionally, HIDENETS and ALARP would have benefited of a fault-tolerant and resilient global time to reduce complexity in distributed algorithms where CSs exchange information along with a timestamp which has to be correctly interpreted. Finally, attentive identification of constraints and governance, and design for evolution would have allowed a deeper contextualization of the problem and produced a long-term vision of the developed systems.

Summarizing, viewpoints like evolution, emergence, governance and constraints are at the core of SoSs and need to acquire more and deeper consideration from system engineers. They together represent one of the most important conceptual instruments to face the escalation of complexity in systems and infrastructures we are witnessing. Thinking of systems in terms of SoSs, starting from their embryonic phase, will ease the systematic design, development and maintenance of mobile safety-critical applications and will support the application of already available proposed solutions by means of the vision brought by the viewpoint-based challenges.

References

[1] M. Jamshidi, Ed. (2009). System-of-Systems engineering - innovations for the 21st century, J. Wiley & Sons.

[2] M. Henshaw et al., "The Systems of Sysems Engineering Strategic Research Angenda," Loughborough University, United Kingdom, TAREA-PU-WP5-R-LU-26, 2013.

[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] Dahmann, J.S.; Baldwin, K.J., "Understanding the Current State of US Defense Systems of Systems and the Implications for Systems Engineering," in 2nd Annual IEEE Systems Conference pp.1-7, 2008.

[6] 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.

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

[8] DYMASOS, Preliminary report on modeling methods, 2013, http://www.dymasos.eu [last accessed 11 December]

[9] Local4Global, Technical Systems of Systems Modelling and Analysis Requirements, 2014, http://local4global-fp7.eu [last accessed 11 December]

[10] COMPASS, Report on Refinement Strategies for SoS Models, 2013, http://www.compass-research.eu [last accessed 11 December]

[11] DANSE (Designing for Adaptability and evolutioN in System of systems Engineering), D4.3 DANSE Methodology V2, http://www.danse-ip.eu/[last accessed 17 December 2014].

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

[13] T. Israr, M. Woodside, and G. Franks, “Interaction Tree Algorithms to Extract Effective Architecture and Layered Performance Models from Traces,” Journal of Systems and Software, vol. 80(4), pp. 474-492, 2007.

[14] K. Nagaraja et al., “Quantifying the performability of cluster-based services,” IEEE Transactions on Parallel and Distributed Systems, vol.16(5), pp. 456-467, 2005.

[15] B. Malinowsky et al., “Timed Broadcast via Off-the-Shelf WLAN Distributed Coordination Function for Safety-Critical Systems,” In Proc. of the 9th European Dependable Computing Conference (EDCC), pp.144-155, 2012.

[16] M. Mock, E. Nett, and S. Schemmer, “Efficient reliable real-time group communication for wireless local area networks,” Proc. European Dependable Computing Conference (EDCC), J. Hlavicka, E. Maehle, and A. Pataricza (Eds.), Springer-Verlag, pp. 380-400, 1999.

[17] TRANSPORT-FP7-234088 ALARP - A railway automatic track warning system based on distributed personal mobile terminals, http://cordis.europa.eu/project/rcn/93402_en.html

[18] A. Ceccarelli et al., “Design and Implementation of Real-Time Wearable Devices for a Safety-Critical Track Warning System,” In Proc. of High-Assurance Systems Engineering (HASE), pp.147-154, 2012.

[19] IST-FP6-26979 HIDENETS - HIghly DEpendable IP-Based NETworks and Services, http://cordis.europa.eu/project/rcn/79303_en.html

[20] A. Bondavalli et al., “The HIDENETS Holistic Approach for the Analysis of Large Critical Mobile Systems,” IEEE Transactions on Mobile Computing, vol. 10(6), pp. 783-796, 2011.

[21] L. Montecchi et al., “Model-based analysis of a protocol for reliable communication in railway worksites,” In the 15th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM) Paphos, Cyprus Island, pp. 23-32, 2012.

[22] D.C. Schmidt, “Guest Editor's Introduction: Model-Driven Engineering” IEEE Computer, 39, pp. 25-31, 2006.

[23] U. Aßmann. “Invasive Software Composition”. Springer, 2003.

[24] U.S. Department of Defense. “Systems Engineering Guide for Systems of Systems”. Version 1.0. Aug. 2008.

[25] 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.

[26] Kopetz H. Real-Time Systems. Second Edition. Springer Verlag. 2011.

[27] E.Morris, P. Place, D. Smith, “System-of-Systems Governance: New Patterns of Thought”, Technical Note CMU/SEI-2006-TN-036, Software Engineering Institute, Carnegie Mellon, October 2006.

[28] “Autonomous Road Vehicles”, U.K. Houses of Parliament, Parliamentaty Office of Science & Technology, POSTnote 443, September 2013.

[29] P. Verissimo, “Travelling through wormholes: a new look at distributed systems models,” SIGACT News 37, 1 (March 2006), pp. 66-81, 2006.

[30] K. Czarnecki and S. Helsen. “Classification of Model Transformation Approaches.” In: Proceedings of the 18th Annual SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and

Applications. OOP SLA'03 (Oct. 26–30, 2003). Anaheim, CA, USA, 2003.

[31] L. Vinerbi, A. Bondavalli, P.Lollini, "Emergence: A new Source of Failures in Complex Systems", Third International Conference on Dependability (DEPEND'10), pp- 133-138, July 2010.

[32] A. Dardenne, A. v. Lamsweerde, and S. Fickas. Goal directed requirements acquisition. In Science of Computer Programming, M. Sintzoff, C. Ghezzi, and G. Roman, editors, number 1-2, pp. 3–50. Elsevier Science, Amsterdam, The Netherlands, Apr 1993.

[33] POR-CREO 2007-2013 Regione Toscana, Project Secure! (2013-2015) http://secure.eng.it [last accessed 12 December 2014].

[34] Simon, H., The Architecture of Complexity. In The Science of the Artificial. MIT Press. Cambridge. 1996

[35] F. Pichler, "Modeling Complex Systems by multi-agent Holarchies," Proc. Of Eurocast'99. P.154, Springer Verlag, 1999

[36] H. Kopetz, "Towards an Understanding of Emergence in a System-of-Systems".

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

[38] H. Kopetz, B. Fromel, O. Hoftberger. "Direct versus stigmergic information flow in systems-of-systems." System of Systems Engineering Conference (SoSE), 2015 10th. IEEE, 2015.

[39] Andrea Bondavalli et al., "Resilient estimation of synchronisation uncertainty through software clocks". *Int. J. Crit. Comput.-Based Syst.* 4, 4 (February 2013), 301-322.

[40] Alwyn Goodloe, and Lee Pike. Monitoring distributed real-time systems: A survey and future directions. National Aeronautics and Space Administration, Langley Research Center, 2010.

[41] M. Dixit et al., "Adaptare: Supporting automatic and dependable adaptation in dynamic environments", In ACM Transactions on Autonomous and Adaptive Systems, Volume 7, Issue 2, Article number 18, pp. 1-25, July, 2012.

[42] Hershey, P.; Rao, S.; Silio, C.B.; Narayan, A., "System of Systems to provide Quality of Service monitoring, management and response in cloud computing environments," System of Systems Engineering Conference (SoSE), 2012 7th. pp.314-320, IEEE, 2012.

[43] J. Joyce, G. Lomow, K. Slind, and B. Unger. Monitoring distributed systems. ACM Trans. Comput. Syst., 5:121–150, March 1987.

[44] CENELEC EN 50126:1999-09. Railway applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS). 1999.

[45] B. Schmerl, J. Aldrich, D. Garlan, R. Kazman, H. Yan, "Discovering Architectures from Running Systems", IEEE Transactions on Software Engineering, vol. 32, no. 7, pp. 454-466, 2006.

[46] A. Ceccarelli, M. Mori, P. Lollini, A. Bondavalli, "Introducing Meta-Requirements for Describing System of Systems" In Proc. of High-Assurance Systems Engineering (HASE)., 2015.

[47] Bondavalli, A.; Ceccarelli, A.; Falai, L.; Vadursi, M., "A New Approach and a Related Tool for Dependability Measurements on Distributed Systems," in IEEE Transactions on Instrumentation and Measurement, vol.59, no.4, pp.820-831, April 2010

[48] H. Moniz, N. F. Neves, M. Correia, and P. Verıssimo. Randomized intrusion-tolerant asynchronous services. In Proceedings of the International Conference on Dependable Systems and Networks (DSN), pages 568–577, June 2006.

[49] N. Medvidovic, and R. Taylor, "A classification and comparison framework for software architecture description languages" IEEE Transactions on Software Engineering, vol. 26, n. 1, pp. 70-93, January 2000.

[50] Object Management Group. "A UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded systems", Version 1.1. OMG Document. June 2011.

[51] M. Mori, A. Ceccarelli, P. Lollini, A. Bondavalli, B. Frömel, "A holistic viewpoint-based SysML Profile to Design Systems-of-Systems" In Proc. of High-Assurance Systems Engineering (HASE), 2016.

[52] S. Biswas, R. Tatchikou, F. Dion, "Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety," Communications Magazine, IEEE , vol.44, no.1, pp.74,82, Jan. 2006.

[53] S. Corbellini, F. Ferraris, M. Parvis, "A System for Monitoring Workers' Safety in an Unhealthy Environment by means of Wearable Sensors," Instrumentation and Measurement Technology Conference Proceedings, 2008. IMTC 2008. IEEE , pp.951,955, 12-15 May 2008.

[54] Y. Lindell, and B. Pinkas, "An efficient protocol for secure two-party computation in the presence of malicious adversaries." Advances in Cryptology-EUROCRYPT 2007, pp. 52-78, Springer Berlin Heidelberg, 2007.

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

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

[57] J. H. Holland, "Emergence: From chaos to order", Oxford University Press, 1998.

[58] R. Abbott, "Emergence explained: Abstractions: Getting epiphenomena to do real work", Complexity, 12 (1), 13-26, 2006.

[59] L. Montecchi, P. Lollini, A. Bondavalli, "A DSL-Supported Workflow for the Automated Assembly of Large Stochastic Models," in Tenth European Dependable Computing Conference (EDCC'14), pp.82-93, 13-16 May 2014.

[60] T. Zoppi et al.. "Presenting the Proper Data to the Crisis Management Operator: A Relevance Labelling Strategy". HASE 2016.

[61] Andrea Bondavalli et al., "Experimental assessment of low-cost GPS-based localization in railway worksite-like scenarios". Measurement, Elsevier (in press), p. 456-466, 2012. ISSN: 0263-2241.

Andrea Bondavalli (M'03) received the M.S. degree in computer science from the University of Pisa, in 1986. He has been a Researcher with the Italian CNR, and is currently a Professor at the University of Florence. He is a Member of the Editorial Board of the IJCCBS journal and the chair of the Steering Committee of the IEEE SRDS. He served as Program Chair and as General Chair of the most important conferences in Dependable Computing and the Conference Coordinator of IEEE DSN 2009.

Andrea Ceccarelli received the Bachelor, and Master degree in computer science, and the PhD in Informatics and Automation Engineering at the University of Firenze, Italy, respectively in 2006, 2008 and 2012. He is currently a Research Associate at the same department. He published in International conferences and journals and regularly serves in the Program Committee of International Conferences and Workshops.

Paolo Lollini is an Assistant Professor at the Faculty of Science at the University of Florence. He has been continuously participating in European funded projects since 2002 up to present. He was a member of the program committee of important conferences in the area and he has coauthored papers that appeared in proceedings of international conferences, journals, and books. His current research interests include the modeling and evaluation of performability and resiliency attributes of large-scale critical infrastructures and systems-of-systems.

Leonardo Montecchi received his Master's degree in Computer Science from the University of Firenze, Italy, in 2010. In April 2014 he received the Ph.D. in Computer Science, Systems and Telecommunications at the same university, where he is currently a post-doc researcher. His main research interests focus on the performability evaluation of complex systems and on model-driven engineering techniques. Often, his research crosses the domains of security and software engineering as well.

Marco Mori received the Bachelor degree and Master degree in computer science from the University of Camerino, Italy, in 2006 and 2008, respectively. He received his PhD from the Institute for Advances Studies, Lucca, Italy in 2012. He is currently a post-doc research fellow at the Mathematics and Computer Science Dept. of the University of Florence, currently working on validating System-of-Systems dependability in the context of the AMADEOS European project. He was previously a two-years post-doc research fellow at the Computer Science Faculty of the University of Namur, Belgium.