ICT
SICS

Proceedings of the Third Swedish Workshop on the Engineering of Systems-of-Systems (SWESoS2018)

Jakob Axelsson (editor)

RISE Report 2018:69

Proceedings of the Third Swedish Workshop on the Engineering of Systems-of-Systems (SWESoS2018)

Jakob Axelsson (editor)

Abstract

Proceedings of the Third Swedish Workshop on the Engineering of Systems-of-Systems (SWESoS2018)

This report contains the proceedings of the third Swedish workshop on the engineering of systems-of-systems, held in Linköping 2018. It contains a brief summary of the event, as well as extended abstracts of six presentations given at the meeting.

Keywords: systems-of-systems

RISE Research Institutes of Sweden AB

RISE Report 2018:69
ISBN: 978-91-88907-18-9
Kista 2018

Content

Preface

Systems-of-systems (SoS) is a topic of increasing importance as the digitalization of society accelerates. The Swedish Workshop on the Engineering of Systems-of-Systems (SWESoS) has the primary purpose of creating a meeting place for researchers and practitioners interested in SoS. The workshop is an informal event, focusing on presentation of results and ongoing research, to stimulate interaction among researchers and practitioners. Following two successful events in Stockholm (2015) and Göteborg (2016), the third workshop in the series was held in Linköping on November 22, 2018. During the three and a half years that have passed since the first event, the interest in the area has increased substantially, and the number of participants has grown from around 20 to around 45.

The program of this year's workshop consisted of a mix of regular papers that were submitted as extended abstracts and reviewed by a program committee, and invited presentations primarily from industry. This document contains the final versions of the regular papers, and the invited presentations were as follows:

- Overview of SoS activities in Sweden and internationally (Jakob Axelsson; Mälardalen University and RISE SICS)
- SoS overview from SAAB, holistic process and challenges (Christopher Jouannet; Saab and Linköping University)
- Towards a harmonized infrastructure for distributed simulation (Björn Möller, Pitch Technologies)
- 5G and Machine Intelligence to Enable SoS (Rafia Iman; Ericsson)
- Keynote: Smart heteogeneous systems (Gunnar Holmberg; Saab and Linköping University)

The interest in the area is also stimulated by research initiatives in Sweden, such as the System-of-Systems for Smart Urban Mobility (SoSSUM) program that has been launched this year by the vehicle research and innovation program (FFI). Due to the importance of this initiative, a special session was dedicated to this program, where one regular paper was included together with the following shorter project presentations:

- SoSSUM: Program overview (Jakob Axelsson; Mälardalen University and RISE SICS)
- ASSET: A System-of-system for Sustainable and Efficient Transport (Győző Gidofalvi; KTH)
- Collaborative systems for patient transports and hospitals (Christofer Englund; RISE Viktoria)
- SMOOTH: System of systems for sustainable urban goods transportation (Else-Marie Malmek; Volvo Group)

Management of Systems of Systems using Goals

Björn Bjurling
RISE AB
bjorn.bjurling@ri.se

***Abstract*—Systems of systems deployed in uncertain and dynamic environments pose hard management challenges due to potentially incomplete, obscured and/or dynamically varying views of the configurability of the component systems. This extended abstract gives a brief account a goal-based management approach for systems of systems that builds upon the framework of policy-based management.**

I. Introduction

Policy-based management regimes are well-established for distributed systems ([1]). It is especially suited for settings where the range of configurability is well-known. However, in contexts with high complexity, stemming for example from uncertainty, resource constraints, and the possibility of federating autonomous systems, policy-based methods can potentially become undesirably ineffective.

This extended abstract describes a selection of potential management issues in such more complex settings and gives a sketch of our approach to the problem. This is done by outlining a class of problems concerning asynchronous and decentralized configuration of federations of heterogeneous and autonomous systems, where the configuration should guarantee some given output or effect through the operations of the federated systems. The problem class covers a broad set of industrial needs that have been addressed in applied research in diverse areas such as: cloud service instantiation; bandwidth management and allocation in ad-hoc tactical radio networks; Effects-Based Operations; self-managing network nodes in telecom; as well as in SLA-based mission planning.

A common theme in the mentioned research efforts is that the desired effect of a service delivered by such systems of systems is well-known, while how, when, where, and by whom the effect should be achieved is known to a lesser extent. Further, in these applications, re-planning or adapting the desired effects are often prohibited. The research has therefore focused on devising suitable and efficient formalisms for expressing desired effects where such expressions define (or more loosely, points to) a maximal set J of possible system configurations, where each such configuration can be assigned a minimal guarantee to lead to some level of the desired effect. Informally, such expressions of desired effects are what we call $goals$ in this context.

II. The System of Systems

In the remainder of this document, the problem class will be illustrated by formulating it in terms of configuration of services in a system of systems setting, while, however, details about the ongoing applied research will be omitted.

Let first, some $n > 0$, $S = (s_0, \ldots, s_{n-1})$, be a system of systems where each system s_i produces services $t_{i,0}, \ldots, t_{i,\ell_i - 1}$ for some $\ell_i > 0$. That a system s_i consumes a service t_{jk} (produced by system s_j) is denoted by t^i_{jk}. Assume for ease of presentation that the configuration of service t^i_{jk} is given by a real number r^i_{jk} in $[0, 1]$ and associate to each service t^i_{jk} a cost $c(r^i_{jk})$ for its configuration. (For example, if s_i and s_j are military units, r^i_{jk} could be the rate at which s_j sends its position to s_i, and the cost could be the bandwidth required for that rate.) In reality, the configuration signature and the cost functions are of course more complex.

Given some configuration for all services t^i_{jk}, we can succinctly describe that configuration as a (potentially rather long) vector:

$$v = (r^0_{00}, r^1_{00}, \ldots, r^{n-1}_{0,\ell_0 - 1}, r^0_{10}, \ldots, \ldots, r^{n-1}_{n-1,\ell_{n-1}-1}).$$

Let $\mathcal{V}$ be the set of all possible configuration vectors v of the system $S = (s_0, \ldots, s_{n-1})$. Then we can associate an *effect* of the system S as a function E from $\mathcal{V}$ to some set O, which we here assume is $\{0, 1\}$—i.e., we assume that a configuration either has or hasn't got an effect.

III. Potential Management Issues

Say for this context that management of the system S amounts to giving an order for a configuration $v \in \mathcal{V}$ for S that achieves a certain effect with minimal cost. Ideally, when the order is the configuration vector v, S gets configured precisely according to v. Due to the nature of the management task, we cannot assume that an order can be precisely implemented. We shall therefore use the notation v' to denote the configuration resulting from the order v. (In the domain of cloud services, the order would instead be called a request).

In the rest of this section, we illustrate some management issues that arise in the setting of industrial application. We give brief hints on how such issues can be addressed in the final section.

A. Dynamics and Uncertainty

If the systems are deployed in the real world there are often factors that affect the configurability of each system. So the order v may result in a configuration v' in some neighborhood $N(v)$ of v. The value of $E(v')$ may differ from the value $E(v)$ expected when the order v was given. Further, due to dynamics, the value $E(v')$ may also change over the time of deployment of the system S.

B. Common and Sparse Resources

For configuring their services, the systems s_i may rely on common resources (such as limited radio bandwidth). Thus the space of feasible configurations is restricted by constraints such as $\sum_{k \in K} r^i_{jk} < L$, for some range K of services that uses common resources and some L. Similar restrictions can be on the output of some service k' at some system i': $\sum_{j<n} r^j_{i'k'} < L'$.

C. Hierarchical Systems

In hierarchical applications, for example in multi-tenant cloud services, otherwise competing service providers may collaborate for delivering a customer service (or *effect*, in our setting). In this situation, individual service providers may be reluctant to disclose the configuration options of proprietary systems, rendering overall system configuration hard to manage. In the tactical domain, an order may have been given long in advance without any knowledge about the status of the system of systems and its configurability at the time of deployment, which may lead to uncertainty about the effect of the order. This issue thus points to a situation where a configuration policy may not even be possible to specify. However in both settings, the order would be better conceived and formulated as a desired effect of the configuration.

IV. Goals for System Configuration

The main instrument we have devised for addressing the above mentioned management issues is that of a goal-based effect request procedure. Without going into detail, we conclude this abstract with an outline of the concept of goals for system configuration. Goals are a kind of instructions that are robust to dynamics and uncertainty in the environment in which the system is deployed, and that can replace the concept of orders w.r.t. functionality. We define four types of goals:

Configuration vectors As an extreme case, a configuration vector v as defined above defines a goal with v as the only possible implementation. This requires that the environment is static and that all the component systems and their configurability are known.

Configuration goals If the systems' services and their configurability are known, we define goals as constraints on the range of configuration parameters. This way the exact configuration can be fine-tuned to fulfill the desired effect while not violating the goal. Configuration goals can suitably be written in the form of SLA:s.

Systems goals When the SoS offers several ways of obtaining an effect, system goals define sets of configuration vectors without specifying which set of component systems should produce the services. Thus, potentially without detailed knowledge about the optimal configuration at the time of authoring the goal, the SoS may fulfill the goal by choosing (through some internal management regime) a configuration that both fulfills the desired effect and (in view of additional information) minimizes the cost for the configuration.

Effect goals When the configurability of the SoS is unavailable, a goal can specify merely the desired effect. It is then up to the SoS to internally manage the configuration of itself to fulfill the effect goal (This can be done using a probabilistic approach as in [2]).

Management via goals means that a SoS should be allowed to dynamically adjust to variations in the configurability as long as the goals it is ruled under are not violated. In case of violation of a goal, and when there is no procedure to adjust the configuration within the given limits, the goal is said to be failed.

Finally, the decentralization of the systems s_i poses high requirements on the system internal management of configurations. Current research is focused on efficient communication between the component systems about their individual resource needs and configuration constraints. Future research directions will include management of conflicting resource needs.

References

[1] Strassner, J., *Policy-Based Network Management* Morgan Kaufmann Publishers, ISBN 1- 55860-859-1, 2004.

[2] Bjurling, B. and Steinert, R. and Gillblad, D.,*Translation of probabilistic QoS in hierarchical and decentralized settings*, IEEE, APNOMS, 2011

Technical Challenges in Designing Systems-of-Systems Supporting Vehicle Fleets

Emmanuel Frécon*, Efi Papatheocharous*
*Software and Systems Engineering
RISE SICS, Isafjordsgatan 22, Kista, Stockholm, Sweden
Email: fname.lname@ri.se

***Abstract*—Advances in interconnectivity between vehicles, vehicle fleets and infrastructures led to opportunities of interoperability and systems-of-systems (SoS). Several challenges emerge that put on requirements on dealing with the vast amount of data generated by modern vehicles and their actuation with higher-level commands and controls. They have naturally created opportunities for the development of sophisticated, powerful, generic platforms to support ingestion, storage, processing, management, operation and orchestration of data and processes in SoS. A prominent example is the scenario of vehicle fleets and more precisely, on how to engineer the SoS so that the collaboration among various constituent systems will achieve the SoS goals. Several challenges cap the extent of opportunities, such as determining the business and functional requirements, as well as technical: constructing and operating an independent, scalable, and flexible platform ensuring e.g., privacy and accountability. In this work, we discuss these concerns and challenges from a technical perspective.**

I. Introduction

If we compare engine technologies used in modern vehicles with other technologies that we use everyday, it seems like not much have changed over the past few decades. What has actually changed drastically in the mobility area, with the drop of component costs and increase of connectivity and computational possibilities in ECUs, CPUs and GPUs[1], is the sophisticated and large number of sensors and actuators, providing information well beyond speed and fuel levels. Further beyond the isolated pieces of information vehicles can produce, the uncapped potential of the value of the combination of those individual data points have to provide to drivers, manufacturers, suppliers, traffic service providers and traffic operators is mostly untapped.

Still, the technologies used in cars (electric, mechanic, computing) are operating quite independently and data is not aggregated to provide extended offerings. Even with the increasing amount of sensors and actuators that will be deployed on future vehicles (mostly driven by the hype of autonomous smart vehicles), without well-thought coordination, cooperation and business strategies, much more useful information is waiting to be exploited and deliver value.

II. Problem statement

An increasing number of vehicles are becoming smart and carry out autonomous activities. Autonomous vehicles are expected to produce up to 4TB per day [1]. When the majority of vehicles are replaced by smart autonomous vehicles and thus increase the number of data generators available, it is critical to discuss the possible actionable outcomes for vehicle operators of fleets and impacts on other infrastructures, but also deal with technical issues such as, bandwidth, latency, reliability, availability, security, safety, scalability and flexibility.

This work, discusses the key technical concerns and challenges met when investigating how to engineer systems-of-systems (SoS) so that the collaboration among various constituent systems can achieve the SoS goals. It is the secondary issue raised by the previous work of Johansson et al. [2] which addressed how to engineer vehicles so to be part of a SoS.

III. Concerns and challenges

Designing SoS for vehicle fleets opens up emergent issues from the collective aggregation of previously existing and individually owned and managed systems. The number of constituents together, as described by Fitzerald et al. [3], offer collectively several functional and technical advantages, and also pose technical and business challenges. Our work focuses on metrics collection, their management and the creation of new services targeting both end-users and professionals in disciplines such as traffic management, traffic services, traffic operation and planning. The composition and design of SoS for vehicle fleets require the deployment of a number of constituents, many of which originate from traditional cloud applications, but are tuned and parameterised to the specific requirements of the heterogeneous and complex systems.

1) Security: Collecting data from remote vehicles and possibly actuate upon these requires a certain degree of openness, but also strong security so that communication cannot be eavesdropped or acted upon by intruders and privacy can be respected. There are a number of intertwined solutions to the construction of a secure system, but key to these solutions is the ability to enforce strong and modern TLS encryption across the entire solution and to keep renewing certificates and keys at regular intervals. Renewals facilitate ageing and obsolescence by automatically and quickly setting aside resources that have not used appropriate means of regeneration. They also improve security by raising the breaching threshold through

[1] Modern vehicles embed an increasing number of Electronic Control Units (ECU), but also more regular Central Processing Units (CPU) and Graphics Processing Units (GPU) for the realisation of, e.g., the infotainment system. Some integrate General-Purpose computing on Graphics Processing Units (GPGPU) to support complex computations inherent to autonomous features.

regular rotation. Examples of such constituents are the use of automatically renewable authoritative certificates through Let's Encrypt[2] or automatic encryption of traffic within the various constituents of the cloud platform as well as to and from the vehicles. Architectures should take a security-by-design approach through a widespread use of these techniques and careful management of sensitive data within and across the platform. In addition, containerisation techniques provide for a high level of encapsulation between the various cloud components, increasing security by making connections and dependencies explicit, traceable and firewalled.

2) Secret Management: Most cloud applications need to exchange secrets between their different components. This should not be confused with authentication: secrets are used at the base of the encryption techniques for security. Secrets are typically blobs of data that should not be transmitted over a network or stored without encryption. Example solutions for its realisation is the concept and implementation of secrets in Docker Swarm[3], or the use of solutions such as HashiCorp's Vault[4]. A security-by-design approach should ensure the use of secrets across the entire platform and service implementation.

3) Deployment and Integration: Modern (web) applications are continuously upgraded and improved. This continuous modification of software has already affected vehicle SoS: while most manufacturers will "update" cars and their subsystem as part of regular servicing, new distribution and actualisation models are trialled by manufacturers such as Tesla [4]. Continuous improvements improve security by quickening the path to security fixes. They also help meeting the end-users' expectations for an improved car-as-a-service experience. Solutions encompass the ability to continuously integrate automatically tested versions of the various software components that are part of the platform and make these available for deployment as soon as possible. This also encompasses solutions for the continuous deployment of (versioned) software components using staging servers or canary deployments[5]. A declarative approach of the infrastructure, such as the one advocated by Machinery[6] provides ways to not only version, branch and merge the code bases of the constituents, but also of the cloud computing architecture itself.

4) Scaling and Elasticity: Collecting metrics from fleets of vehicles requires flexibility: in time an increasing number of cars will be part of the SoS, and the number of cars sending (and receiving) data will change at various time of the day and/or within different geographical regions. Containerisation techniques and the widespread use of the "pets vs. cattle" model[7] across the cloud infrastructure are key to the realisation of architectures that are able to adapt to these dynamic variations across time and space. Orchestration tools such as Docker Swarm[8] or Kubernetes[9] makes it easy to let the amount of container replicas fluctuate over time and with demand. In general, solutions should carefully select constituents that have a known migration path to horizontal scalability and high availability. In addition, the declarative approach provided by tools like Machinery or Terraform[10] facilitates cloud providers independence and hybrid cloud deployments for improved resilience to failures. From a SoS perspective, these properties bring an increased flexibility and the realisation of self-healing systems, lowering risks for failures and increasing reliability.

5) Streamlining Data: Using standardised streamlining, processing and utilisation services makes it easier to unlock the value from the vast quantity of data being generated by vehicles. This involves the use of Open APIs for data access from external services, but also the use of standardised data and communication protocols to facilitate connection to/from hybrid fleets, e.g., across models and manufacturers. For example, PostgREST[11] easily turns any PostgreSQL [6] database directly into a RESTful API. This interface is application-independent and generic, and it facilitates access to the underlying database from modern back-end services or devices. While MQTT [7] provides a standard for loosely decoupled communication between interested parties, it leaves several open design decisions, such as topics organisation and data formats. However, constituents can turn to standards like SenML [8] serialised to JSON as a common data streaming format. Integrating further formats, such as the CoRE link format [9], could provide an embryo to a standardised management of devices and their capabilities. From a SoS perspective, this facilitates the provision of emergent properties by enabling recurrent systems integration, e.g., further integrating with external systems through the fusion of several data sources.

6) Highly Available Databases: Collecting metrics from fleets of vehicles requires the deployment of time-series databases, sharding of data and databases capable of expressing spatial queries. As the amount of data grows with the number of sensors and complexity of vehicles, solutions should also address where data analysis should take place and at which abstraction level information should be stored. As embedded computing power increases, offloading some data analysis to the vehicles can be key to reduce the volume of cloud storage required. For cloud storage, time-series databases [10] such as InfluxDB are often picked for their specificity. However, extensions to regular SQL databases such

[2] Let's Encrypt (https://letsencrypt.org/) is a free, automated, and open certificate authority from the non-profit Internet Security Research Group (ISRG).

[3] Docker Secrets are described in "Manage sensitive data with Docker secrets", available at https://docs.docker.com/engine/swarm/secrets/.

[4] Vault is documented at "Learn about secrets management and data protection with HashiCorp Vault", available at https://learn.hashicorp.com/vault/.

[5] A canary deployment is the push of changes to a smaller group of users or vehicles without their knowledge, they facilitates testing of features in real conditions without adventuring the entire installed base.

[6] Docker Machinery is an open-source cluster management tool developed by one of the authors, available at https://github.com/efrecon/machinery.

[7] The "pets vs. cattle" [5] model captures the concept that servers and other resources are dispensable and can be destroyed and removed at any time.

[8] The "Swarm mode overview" provides a good entry point to concepts and features behind Swarm, available at https://docs.docker.com/engine/swarm/.

[9] Kubernetes (https://kubernetes.io/) is an open-source system for automating deployment, scaling, and management of containerised applications.

[10] Terraform (https://terraform.io/) enables to write, plan, and create infrastructure as code.

[11] Full PostgREST documentation available at https://postgrest.com/.

as Timescale[12] bring a common solution to all data storage and querying requirements, thus promising to reduce system complexity and the necessity for specialised databases. From a SoS perspective, this facilitates the creation of applications requiring large quantities of data points to refine their reasoning over time and space (e.g., Big Data, Artificial Intelligence).

7) Data Backup: Ensures access across time and resolution to a previous state in the case of incidents. Solutions should include off-site storage and the development, routine testing and documentation of backup solutions in order to facilitate recovery. A recent incident [11] at GitHub has shown the importance of these backups, as the entire database had to be recreated from backups to revert to a stable state. With a declarative approach to software and infrastructure, the state of various constituents is programmable and deterministic. Independent software backups are less important to system integrity as there is often a clear path to resilience from a known base-level state.

8) Supervision and Auditing: Modern applications and fleet supporting SoS evolve continuously, and even host varying versions of underlying constituents. In such systems supervision, together with log collection and analysis become key to stability and the ability to understand and solve problems when they appear. Solutions should involve subsystems that are deployed and maintained in isolation from the remaining constituents or external services. This will improve reliability and ensure continuity in the advent of problems that might perturb or totally impair normal operation. This separation of concerns is critical to being able to communicate on-going status reports to end-users [11].

IV. CONCLUSION AND FUTURE WORK

In addition to the technical concerns raised in this paper, a number of related business and functional concerns exist. For instance, the need of developing business strategies for managerial and operational independence [12], which has implications on safety in vehicle fleets as discussed in [13]. Moreover, a necessary requirement is creating new business models and ways of working within the complex socio-technical system involving both humans and machines interacting with each other [13]. Dealing with issues like IP management, agreements, participating in cooperations (ecosystems), cost sharing, data sharing and profit sharing and coordination of efforts in standardisation [12]. Functional examples include for the stakeholders coming from different organisations and having different interests to be able to openly innovate and co-create value, carry out data-driven decision making, use semantics, annotations and models.

In this paper, we focused on the technical concerns and challenges when investigating how to engineer SoS so that the collaboration among various constituent systems will achieve the SoS goals. Our work plans to support the need of a conceptual architecture of a generic platform to enable data-driven SoS for vehicle fleets based on the technical challenges mentioned above. As a preliminary proof-of-concept, we have developed a demonstrator which supports two-way connectivity needs of third-party cloud services for quantified connected vehicles [14]. The demonstrator will be extended in the future to include aggregated decision making support infrastructures for traffic management centres, traffic service providers, traffic operators and municipalities.

ACKNOWLEDGMENT

Part of the work is supported by the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737422. This Joint Undertaking receives support from the European Unions Horizon 2020 research and innovation programme and Austria, Spain, Finland, Ireland, Sweden, Germany, Poland, Portugal, Netherlands, Belgium, Norway.

REFERENCES

[1] B. Krzanich. (2016) Data is the new oil in the future of automated driving. Accessed: 2018-10-25. [Online]. Available: https://newsroom.intel.com/editorials/krzanich-the-future-of-automated-driving/

[2] E. Johansson, T. Larsson, M. Aramrattana, P. Pelliccione, M. Agren, G. Jonsson, and R. Heldal, “Systems of systems concepts for cars,” 2017.

[3] J. Fitzgerald, P. G. Larsen, and J. Woodcock, “Foundations for model-based engineering of systems of systems,” in *Complex Systems Design & Management*. Springer, 2014, pp. 1–19.

[4] S. Vöst and S. Wagner, “Towards continuous integration and continuous delivery in the automotive industry,” *CoRR*, vol. abs/1612.04139, 2016. [Online]. Available: http://arxiv.org/abs/1612.04139

[5] R. Bias. (2016, Sep.) The history of pets vs cattle and how to use the analogy properly. Accessed: 2018-10-25. [Online]. Available: http://cloudscaling.com/blog/cloud-computing/the-history-of-pets-vs-cattle/

[6] PostgreSQL Global Development Group, “PostgreSQL,” http://www.postgresql.org, 2018, Accessed: 2018-10-25.

[7] “MQTT Version 3.1.1 Plus Errata 01,” OASIS Open, Tech. Rep., Dec. 2015, Accessed: 2018-10-25. [Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html

[8] C. Jennings, Z. Shelby, J. Arkko, A. Keranen, and C. Bormann, “Sensor measurement lists (SenML),” IETF, memo 15, May 2018, Accessed: 2018-10-25. [Online]. Available: https://tools.ietf.org/html/draft-ietf-core-senml-15

[9] Z. Shelby, “Constrained RESTful Environments (CoRE) Link Format,” IETF, RFC6690, Aug. 2012, Accessed: 2018-10-25. [Online]. Available: https://tools.ietf.org/html/rfc6690

[10] A. Bader, O. Kopp, and M. Falkenthal, “Survey and comparison of open source time series databases,” in *Datenbanksysteme fr Business, Technologie und Web (BTW 2017) - Workshopband*, B. Mitschang, D. Nicklas, F. Leymann, H. Schning, M. Herschel, J. Teubner, T. Hrder, O. Kopp, and M. Wieland, Eds. Bonn: Gesellschaft fr Informatik e.V., 2017, pp. 249–268.

[11] J. Warner. (2018, Oct.) October 21 post-incident analysis. Accessed: 2018-10-25. [Online]. Available: https://blog.github.com/2018-10-30-oct21-post-incident-analysis/

[12] M. W. Maier, “Architecting principles for systems-of-systems,” *Systems Engineering: The Journal of the International Council on Systems Engineering*, vol. 1, no. 4, pp. 267–284, 1998.

[13] A. Kobetski and J. Axelsson, “Towards safe and secure systems of systems: challenges and opportunities,” in *Proceedings of the Symposium on Applied Computing*. ACM, 2017, pp. 1803–1806.

[14] E. Papatheocharous, E. Frécon, C. Kaiser, A. Festl, and A. Stocker, “Towards a generic IoT platform for data-driven vehicle services,” in *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. IEEE, 2018, pp. 95–100.

[12] Timescale (https://www.timescale.com/) is an open-source time-series database compatible with PostgreSQL for fast ingest and complex queries.

Scaling System-of-Systems by Open Self-Organizing Solutions

Richard Bunk
School of IT, Halmstad University, Combitech
Halmstad/Göteborg, Sweden
richard.bunk@hh.se

Magnus Bergquist
School of IT, Halmstad University
Halmstad, Sweden
magnus.bergquist@hh.se

Dulce Goncalves
Combitech, School of IT, Halmstad University
Göteborg/Halmstad, Sweden
dulce.goncalves@combitech.se

***Abstract*—This paper reports from the ongoing Vinnova project VirtualCargo that aims to use a systems-of-systems approach to develop a MaaS for unlimited and unforeseen suppliers, types of services, users, modality shifts, modes of transportation, and demands.**

Keywords—MaaS platform, systems-of-systems, self-organizing systems

I. Background

Development goes towards a society where people increasingly use services to solve their transport needs, which has created a societal demand for more sustainable ways of consuming resources. More people live in cities, commute and travel in work and leisure. This creates more and more complex travel patterns, and we observe a strong trend towards shared economies.

II. Problem to address

Mobility as a Service (MaaS) has developed into a powerful concept for future point-to-point mobility using a systems approach to multimodal travels [1]. Some MaaS solutions are in fact integrating several systems into system-of-systems: an integrating (digital) platform functions as a hub in an ecosystem of service providers and customers who pool resources and capabilities with other systems, that in turn generate new and even more complex system-of-systems with functionality and performance that transcends the sum of the inherent parts.

Still, most such solutions are limited in their ability to scale. The reason is that they are designed for one or few service providers with a fixed set of specified services made available within a geographically limited space [2]. This limitation will severely hamper future MaaS ability to provide services for increasingly complex multi-modality travel patterns with high variety of unforeseen user needs. It is nearly impossible to identify and anticipate all future transportation needs and service offerings across all transport modalities, demographics, locations, time periods, trends and changing individual needs. Existing platforms are not designed to scale volume and complementing services to solve dynamic multimodal mobility and adapt to unlimited service providers and changed customer needs in a flexible way.

III. Project goal

The goal with our project, *VirtualCargo* is to develop an open and self-organizing Mobility as a Service (MaaS) platform. We attempt an entirely different approach than previous solutions that are trying to explicitly specify all such variants ("developer-driven hard design"). We will develop a MaaS for *unlimited and unforeseen* suppliers, types of services, users, modality shifts, modes of transportation, demand, etc.

Our design approach is:

- A platform that is designed bottom-up to support an open eco-system of suppliers as well as end-users/consumers (travelers).
- A platform that will contain mechanisms to allow self-organization of the eco-system so that systems can add to systems.
- Design mechanisms that will be based on demand and supply on the market (*"market-driven adaptive design"*) – no matter where in the world this is taking place.

IV. Expected effects

The proposed solution will in principle not have to be explicitly designed for custom-feature for any service supplier or consumer use-case. It will support an unlimited number of suppliers, service offerings and developers, with tight customer integration. New types and combinations of service offers will be able to spontaneously take shape through market-driven expectations and business-driven transport service providers. Even peripheral services, not primarily tied to the actual transportation, will be able to take shape and enter the eco-system, such as hotels, restaurants.

Specifically, the solution will:

- Improve customer facing, and customer experience with new products/services. Our key objective is to increase revenue per product/service, and revenue per customer.
- Create new business models. Key objective is to generate new revenue from new customers
- Business operations. Key objective is to increase operational efficiency and flexibility
- Developer experience. Key objective is increased developer productivity

V. Three sub-projects converging

The project uses a Service Design approach [3] to integrate three crucial aspects of MaaS design: *Business*

Model Innovation, *UX Design*, and *Technology Design*. Together, these create a digital-platform synergy.

VI. Ambitions and expected output

The platform will be a real working product (Minimum Viable Product, MVP). Our plan is to launch this MVP to real-world users at an early stage so that early adopters can provide us with valuable feedback, i.e. Lean Startup model. This feedback data will both serve as input for research studies and material to complete several iterative product improvements.

VII. Preparations and startup

A workshop series is planned for this autumn. This will include all industrial partners as well as researchers, and aim to set the scope and procedures for our collaboration and co-creation of the product. Here, we will also identify any further partners that we would like to connect to the project. A suitable product operator will be decided on collectively. This is important to reach maximum efficiency and also pave the way for commercial neutrality. A Product Owner at the Future Mobility Center (FMC) will be chosen by the project participants.

Final comments

The project will look into possibilities to build upon pre-existing tools. For instance, we will investigate how solutions such as Google Maps can be used to advice users how to get from A to B. We will also investigate what nodes for connecting people across various transport modalities are already available. existing API:s can be built upon. We see "predictability" as one of the most important features of successful MaaS solutions. This will be increasingly important, and is gradually becoming a business enabler.

References

[1] S. Hietanen, "'Mobility as a Service' – the new transport model?" *Eurotransport*, vol. 12, 2014, pp. 2-4.

[2] P. Jittrapirom, V. Caiati, A.-M. Feneri, S. Ebrahimigharehbaghi, M. J. A. González, and J. Narayan, "Mobility as a Service: A Critical Review of Definitions, Assessments of Schemes, and Key Challenges," *UP*, vol. 2, no. 2, pp. 13–25, Jun. 2017.

[3] R. F. Lusch and S. Nambisan, "Service innovation: a service-dominant logic perspective," *MIS Quarterly*, vol. 39, no. 1, pp. 155–175, Jan. 2015.

The project VirtualCargo is funded by Vinnova call "Systems-of-Systems for mobility in cities (SoSSUM) - FFI - 2018-06-12".

Future autonomous airports: a system-of-systems approach

Lei Chen[1] and Cristofer Englund[1,2]

Abstract— **With the increasing automation and connectivity in the airport industry, airport functions are becoming increasingly automated. In this paper, we present and discuss future autonomous airports where the majority of the ground functions are automated with autonomous vehicles and machines. We present the conceptual system, a case study, and aim at approaching the system engineering with the system-of-systems (SoS) methodology.**

I. INTRODUCTION

Airport operational system is a complex system with many independent systems provided by different vendors and running on shared and/or independent infrastructure. Some systems are more tightly related while others are loosely related. For those systems that need coordination, ad-hoc integration exists where point-to-point coordination is mostly involved, while a holistic framework for system integration to maximize the operational efficiency is yet to be developed.

Improving the operational efficiency at airports has enormous benefits. The forthcoming emerging technologies such as connectivity, autonomous vehicles, and artificial intelligence will contribute to automate airport operations. It can be expected that many sub-systems can be automated or even completely autonomous. Those systems are independent system but need to coordinate with each other or with other airport functional systems to enable efficient airport operation, forming a system of systems (SoS), i.e., Autonomous AirPort SoS (AAP-SoS).

In this work, we present the concept of future autonomous airports and take our on-going work at Örnsköldsvik Airport (OER) as a case study. We discuss the main automated functions, the alignment of SoS, and a preliminary case study.

II. AUTONOMOUS AIRPORTS

We envision future autonomous airports where operational functions are highly automated or fully autonomous. Air traffic control has already been centralized through Remote Tower Services (RTS) such as the one introduced by LFV [1]. As from 2016, driven by the project Digital Run Way Incursion Warning Systems (DRIWS) [2], geo-fencing has been introduced in the Swedish airports, where physical traffic lights have been replaced by digital traffic signals [3]. With the on-going automation efforts, other functional areas will soon be automated. A small scale trial focusing ground vehicle operations is under investigation at the OER airport as illustrated by Fig.1 and described as follows.

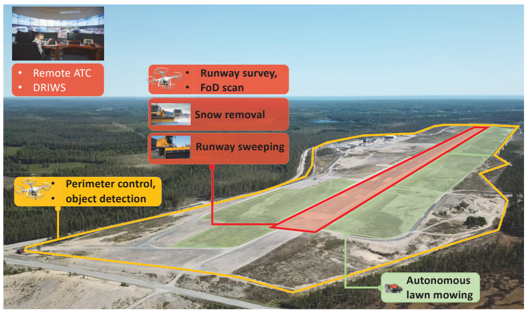

Fig. 1. Future autonomous airports: ground operations

- **Remote ATC (R-ATC)**: With the introduction of RTS, ATC has been moved to remote towers, thus we consider a AAP-SoS with remote ATC. All ground vehicle movement need to coordinate with R-ATC.
- **DRIWS**: DRIWS is the geo-fencing system that is integrated with R-ATC for controlling the access of runway as shown by the colored zones in Fig. 1.
- **Autonomous perimeter control**: Perimeter control is the routine task and labor intensive task for controlling the airport perimeter for identifying potential intrusions. We employ drones for automating those tasks.
- **Autonomous FOD detection and pavement survey**: Foreign object debris (FOD) include many objects that do not belong to the places they are found. Runway pavement needs to be kept in a good condition to ensure safety. We employ drones for FOD scan and pavement survey. **Autonomous runway sweeping** may be initialized directly if necessary.
- **Autonomous snow removal**: Snow removal is a heavy task, especially when considering the time limit. **Autonomous snow removal** consists of a platoon of snow removal vehicles that run in a coordinated way, significantly improving the operational efficiency.
- **Autonomous runway sweeping**: Runway sweeping is similar to snow removal but need to be done all around the year. The tasks can be scheduled when there is no air traffic or during night.
- **Autonomous lawn mowing**: Maintaining the airside lawn is another labor-intensive task. Autonomous lawn mowers are employed to execute those tasks.

Those are the automated systems that are under consideration in the first phase of autonomous airports. With testing and continuous use case development, it is expected

[1] Lei Chen and Cristofer are with Research Institutes of Sweden, ICT Viktoria, Lindholmspiren 3A, SE 417 56, Gothenburg, Sweden lei.chen, cristofer.englund at ri.se

[2] Cristofer Englund is also with Center for Applied Intelligent Systems Research (CAISR), School of Information Technology, Halmstad University, SE 301 18, Halmstad, Sweden

that more automated systems will be included. With also integration towards other ground handling systems and air traffic control systems, an overall AAP-SoS surfaces.

III. Autonomous Airports as a System of Systems

Autonomous airports is a SoS that satisfies the SoS properties [4] and need to be approached from the perspective of SoS during all phases of the design, modeling, simulation, testing, verification, and deployment.

- **Operational independence**: All above listed automated systems operate independently with missions defined in their own domains. Together with all other airport systems, they form an autonomous airport system that is part of the whole air transportation eco-system. For example, **autonomous FOD detection** and **autonomous pavement survey** can operate without any coordination with e.g., **autonomous runway sweeping**, however, initializing **autonomous runway sweeping** automatically after the detection of FOD on the runway is favorable to maximize the safety.
- **Managerial independence**: All above listed automated systems may be provided and governed by different suppliers that may or may not have any business interactions. Even for the cases of drones, **autonomous perimter control** and **autonomous pavement survey** may be delivered by different companies with different drones dedicated for specific tasks.
- **Evolutionary development**; The development and existence of the listed automated systems are evolutionary. They may be added with new functionality or modified depending on the airport requirements and operational results. For example, **autonomous perimeter control** may start with video streaming to airport staff and basic detection algorithms. They can then be equipped with powerful on-board computers for streaming data analysis and activity classification such as human detection, intrusion detection. They may even have capabilities to execute certain tasks locally to e.g., warn the intruders.
- **Emergent behavior**: The autonomous airport functions can not be executed by any of the single automated systems listed above and thus the airport behavior can not be predicted or realized by any of the single automated systems. The emergent behaviors could be a result of the potential relations among each system and may be exposed during the AAP-SoS evolution.
- **Geographic distribution**: AAP-SoS systems may be geographically distributed due to the introduction of remote tower services and remote vehicle/machine control. Since the introduction of RTS, ATC has been moved from the airport to remote towers. Since DRIWS is tightly integrated with ATC, it is also located at a different geographical location. The drone operating center may be located at other places such as the control center of the service providers.

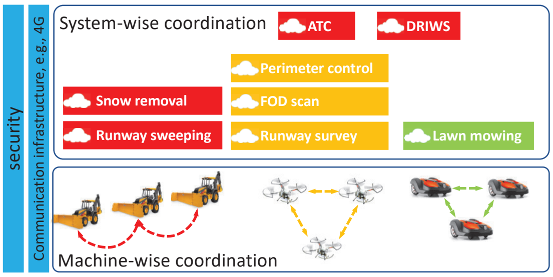

Fig. 2. An AAP-SoS architecture

IV. A Case Study

A preliminary system design now is under development for the AAP-SoS as shown in Fig. 2. The ground level are vehicles and machines, where vehicle-to-vehicle (V2V) or machine-to-machine (M2M) communications may be considered for local coordination. The upper level illustrates the system-wise coordination. Systems are independent but need to communicate with each other to fulfill the overall airport operation requirements.

R-ATC and DRIWS are remote systems that controls air traffic and the access of different areas within the airport. DRIWS defines Geo-zones for the airside access and all ground vehicles and machines need to strictly follow the Geo-fencing rules for executing their tasks. For each of the individual systems, they are mostly provided by different vendors and have different control or planning systems. For example, **autonomous snow removal** is provided by Yeti [5], **autonomous perimeter control** is provided by FlyPulse together with RISE [6], and **autonomous lawn mowing** is provided by Husqvarna. Each individual system is independent but need to coordinate with each other for overall airport operations.

V. On-going Works and Challenges

While each stakeholder focuses on the design and development of their own systems, we aim at an overall SoS design that reaches a balance between sub-system optimization and AAP-SoS goals to maximize operational efficiency. From the SoS design perspective, the project is at a very early age to formulate the architecture of AAP-SoS. Together with all stakeholders and using the current pilot as a case study, we aim at developing processes that can assist the development of AAP-SoS for future autonomous airports.

There are many challenges for general SoS engineering regarding all stages from modeling, architecting, to simulation, testing and verification [7]. Modeling and architecting of SoS need to capture all aforementioned SoS properties. Model-based SoS engineering seems promising while further investigation is needed. Simulation is important for analyzing the system, and developing a SoS simulator that captures all SoS properties is extreme challenging. There have been many different strategies including agent-based simulation, high-level architecture (HLA) based simulation, as well as general purpose simulation. They need to be investigated for the application for AAP-SoS with also consideration and utilization of available airport simulating platforms. SoS testing needs to capture potential deviations as the composed system may not fulfil the expected emergent properties. Testing methods can be found in many standards while analysis has to be done together with airport operators to identify a proper testing method. Verification of SoS is rather a new area where new theories and tools are needed to e.g., proof the correctness of SoS.

Regarding the above challenges and the application of SoS in an airport environment, there have been common practices, standards, and methodologies. A first step is to go through the available SoS engineering methods and successful use cases. In the meanwhile, since airports have their own systems, introducing new systems will need their close engagement. Discussion have been initialized with identification of certain topics to consider while further and deeper analysis are expected.

VI. Conclusions

We present future autonomous airports for improving operational efficiency and reducing operational costs with a preliminary case study. We describe the initial system design with alignment to the SoS properties and aims at applying SoS methodologies to the system design and development.

References

[1] LFV Remote Tower Services, https://goo.gl/xYNyQs.

[2] Digital Runway Incursion Warning Systems, https://www.viktoria.se/projects/driws.

[3] Englund, Cristofer and Didoff, Jonas and Wahlström, Björn: A new method for ground vehicle access control and situation awareness: experiences from a real-life implementation at an airport, Proc. 24th World Congress on Intelligent Transportation Systems, 2017

[4] Maier, M.W.: Architecting Principles for System of Systems, Systems Engineering, Vol . 1, No. 4, 1998, pp. 267-284.

[5] Autonomous snow removal, https://goo.gl/wMYm1k.

[6] Airport Surveillance for Airport Safey, https://goo.gl/cqBZCo.

[7] laus Ballegaard Nielsen, Peter Gorm Larsen, John Fitzgerald, Jim Woodcock, and Jan Peleska. 2015. Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions. ACM Comput. Surv. 48, 2, Article 18 (September 2015)

Towards a System-of-Systems Architecture for Construction Applications Based on Industry 4.0

Jakob Axelsson
Mälardalen University and RISE
Västerås, Sweden
jakob.axelsson@mdh.se

Joakim Fröberg
RISE Research Institutes of Sweden
Västerås, Sweden
joakim.froberg@ri.se

Peter Eriksson
Volvo Construction Equipment AB
Eskilstuna, Sweden
peter.eriksson.2@volvo.com

***Abstract*—The efficiency improvements in the road construction sector during the last few decades have been negligible, whereas other industries have seen very large improvements by applying automation and Lean-based flow optimization. In this paper, we outline concerns and principles for a flexible and extensible system-of-systems architecture for road construction aiming at closing this efficiency gap. It adapts key ideas from Industry 4.0, such as hierarchical decomposition, common interfaces, and ontologies.**

Keywords—system-of-systems; road construction; Industry 4.0.

I. Problem Description

The construction sector is one of the largest industries in the world, with an annual global turn-over of around 13% of the global GDP [1]. However, whereas other industries such as manufacturing have seen improvements in the order of 3.6% per year over the last 20 years, the improvement rate in construction is only about 1% per year [1].

In our research, we make a hypothesis that this gap is in part due to lack of communication and coordination between the parties involved in construction, and that a system-of-systems (SoS) approach can be fruitful.

In a previous paper, we have identified improvement potentials in road construction [2], which is in itself a significant sector contributing about 1% of Sweden's GDP. One of the findings was that the SoS architecture is a key enabler, and in this paper, we extend previous research to outline the key concerns and principles of such an architecture.

II. Research Approach

This paper addresses the following research question: "What is a suitable system-of-systems architecture to support efficiency improvement in road construction?" The research is thus solution oriented, and we apply a design science research method [3], where we use patterns from the existing knowledge base, and gather needs and perform validation using our industrial partners, Volvo CE and Sandvik.

The artefact in focus is an architecture description, but for validation purposes we have in parallel also developed a prototype implementation whose purpose is to show that the key architectural elements of the SoS architecture can interact as intended, and that the anticipated benefits are within reach.

Since the manufacturing industry sets the efficiency mark, an important part of the knowledge base is techniques used there, including Lean [4] and Industry 4.0 [5].

III. Architectural Concerns and Principles

The construction SoS has several architectural concerns, which do not always align with those used in other industries. The key concerns were elicited through workshops with domain experts, and include:

- *Multi-vendor*. Machines from different vendors and of different types must be able to collaborate on the construction site.
- *Autonomous and manual*. Current road construction equipment is mostly manually operated, but there is a strong trend to develop more automated solutions. The SoS architecture must thus be able to handle both types, and a mix of them.
- *Secure*. Participating in an SoS requires a certain degree of openness, and it must be ensured that the communication interfaces do not allow manipulation. It must also be assured that confidential information of a certain participant does not become accessible to others.
- *Flexible*. A difference between road construction and manufacturing is the continuous changes in the former. The process has much shorter periods of steady state, which makes process optimization more difficult. This increases the need for up-to-date information, support for re-planning and reconfiguration. The variability between different construction projects is substantial.
- *Robust*. It cannot be assumed that communication is reliable all the time, since road construction must rely on wireless communication, and the coverage of cellular networks is often poor.

Based on these concerns, some key architectural principles have been identified, which are based on similar ideas as used in the Reference Architecture Model for Industry 4.0 (RAMI4.0) [5]:

- *Asset administration shell (AAS)*. To provide a common interface to constituent systems, RAMI 4.0 introduces the AAS concept, which can encapsulate an asset such as a physical machine and give it proper information interfaces. This allows for different assets to communicate in a standard way and provides mechanisms for self-description. Note that the AAS may also provide an interface to a human operator, thus catering for the need to handle both autonomous and manual machines.

This research was funded by Vinnova, Formas and Sweden's Energy Agency, within their joint program InfraSweden2030, under grant no. 2018-00671.

- *Hierarchy*. Construction work is today organized in a hierarchy, where the working machines are at the bottom. The next layer is the work site (or a part within it). Above that is a project level, which can coordinate several sites (e.g. a road site, a quarry, and an asphalt plant). However, this structure is in fact usually a poly-hierarchy, where certain parts can serve several parents simultaneously. The different parts in the hierarchy are usually ran by different organizations, resulting in an operational and managerial independence. In our proposed approach, elements on all levels are treated as assets, and given their own AAS to handle interactions.

- *Capabilities and sub-models*. The different assets are described in terms of their capabilities, i.e. what services they can provide. For each capability, there is a sub-model that implements the service, making the design of the constituent systems modular. Capabilities include the ability to use different communication techniques, but also different physical work that can be done depending on the machine type.

- *World model and ontologies*. Each constituent system of the SoS will contain a substantial amount of information about other constituents, as well as data about the environment they operate in. We call this information set its world model, and at times it is essential to extract data from the world model and exchange it with other assets. To solve this interoperability problem in a flexible and extensible way, the world model and communication use linked data as defined in the semantic web initiative [6]. A part of the world model is also ontological information describing the relations between key concepts.

- *Communication*. Within a sub-process in the hierarchy, the involved AAS's may communicate either point-to-point using short-range radio or through cellular Internet connections, thus increasing robustness by providing alternative paths. By joining an AAS to a sub-process, it is also given the credentials to communicate in that context, and this is one of several measures to deal with security.

IV. Conclusions and Future Work

In this paper, we have presented findings related to an SoS architecture that can support efficiency improvements in the construction domain. A first version of the architecture description is near-ready, and undergoing validation using a prototype implementation coordinating simulated physical assets. The next steps in this work are to extend this simulation with a wider range of different machines and capabilities, and to extend the analysis and optimization support in order to quantify the reachable level of efficiency improvement. After this, a proof-of-concept demonstrator is planned, using real machines at a site, allowing to evaluate interactions with operators and real efficiency improvements.

References

[1] McKinsey & Co. “Reinventing Construction: A Route to Higher Productivity.” Feb. 2017.

[2] J. Axelsson, J. Froberg, and P. Eriksson, “Towards a System-of-Systems for Improved Road Construction Efficiency Using Lean and Industry 4.0,” in *Proc. 13th Annual Conference on System of Systems Engineering (SoSE)*, 2018, pp. 576–582.

[3] A. R. Hevner, S. T. March, J. Park, and S. Ram, “Design science in information systems research,” MIS Q., vol. 28, no. 1, pp. 75–105, 2004.

[4] P. Hines, N. Rich. "The seven value stream mapping tools." Intl. J. of Operations and Production Management, vol. 17, no. 1, pp. 46-64, 1997,

[5] DIN SPEC 91345. Reference Architecture Model Industrie 4.0 (RAMI4.0). April, 2016.

[6] T. Berners-Lee, J. Hendler, and O. Lassila, “The Semantic Web,” *Scientific American*, vol. 284. pp. 34–43, 2001.

Defining a Method to Perform Effective Hazard Analysis for a Directed SoS Based on STPA

Stephan Baumgart[*], Joakim Fröberg[†‡], Sasikumar Punnekkat[‡]
[*] System Architecture Department, Volvo Construction Equipment, Eskilstuna, Sweden
Email: stephan.baumgart@volvo.com
[†] Research Institutes of Sweden, RISE ICT/SICS, Sweden
Email: joakim.froberg@ri.se
[‡]School of Innovation, Design and Engineering, Mälardalen University, Västerås, Sweden
Email: sasikumar.punnekkat@mdh.se

***Abstract*—Automating a quarry site as developed within the electric site research project at Volvo Construction Equipment is an example of a directed system-of-systems (SoS). In our case automated machines and connected smart systems are utilized to improve the work-flow at the site. We currently work on conducting hazard and safety analyses on the SoS level. Performing a hazard analysis on a SoS has been a challenge in terms of complexity and work effort. We elaborate on the suitability of methods, discuss requirements on a feasible method, and propose a tailoring of the STPA method to leverage complexity.**

***Index Terms*—Hazard Analysis and Risk Assessment, System-of-Systems, Autonomous Machines, STPA, Safety**

I. Safety analysis for SoS

We are currently working with safety analysis of an intended automated quarry, and the objective of this paper is to present our approach with using STPA and elaborate on how to define an effective method to perform hazard analysis in a system-of-systems (SoS).

Safety and hazard analysis methods such as Preliminary Hazard Analysis (PHA), Failure-Mode and Effects Analysis (FMEA) and Fault Tree Analysis (FTA) [1] are well established in industry and these methods are required by functional safety standards. Today's industrial development processes are often tailored to develop single systems, where the intended operational context is used as an important input when analyzing potential hazards. As single systems get more automated and connected to its surrounding world, their behavior becomes more advanced. They become smarter in the sense that they exhibit more functionality and thus safety analysis takes a larger effort.

When products are connected to form a SoS, their ability to interact and share services and signals to achieve cooperative goals need to be explicitly addressed. The usage may deviate from what was intended for a single product. Interactions and emergent behavior in a SoS can give rise to hazards and unsafe work environment although each system in itself is already analyzed thoroughly. The application of standard hazard and safety analysis methods for analyzing a SoS may not be enough and it may not be the most efficient method.

II. Industrial Case - An Automated Quarry

The automated quarry site in our case is operated with machines and other systems that are cooperating to meet goals of productivity and quality, but also a safe and hazard-free work environment. Many of the constituent machines are highly automated and are connected to off-board systems to monitoring of the production process. In our case, the quarry is a surface mine with different production stages, where material is transported by haulers between production steps for further processing. In the electric site research project [2], the work-flow at the site is adapted by using automated haulers, called HX, for material transporting purposes. The HX machines operate in a fleet and are track-based automated guided vehicles (AGVs) (Fig. 1) [3], which receive their work-missions from a fleet control system. Knowing the correct position of all involved machines is necessary for executing missions and avoiding accidents.

An excavator loads a crusher that loads crushed rocks directly onto semi-automated haulers that, in turn, transport and tip the material to a secondary crusher. The operation is supported by several information systems. A site management system is operated by a site operator to monitor production and tune the production process. Machines are also connected to maintenance and fleet management systems. Some machines are equipped with positioning systems.

The site system is an example of a system-of-systems where different smart systems are independent in terms of management, ownership and life-cycle. This is a directed SoS [4] and the constituent systems use their abilities to cooperate to achieve production in the quarry. Furthermore, the constituent systems use smartness to optimize, e.g., machine wear or energy consumption, giving rise to emergent behaviors. But some emergent behavior could be unwanted or even associated with risk. Constituent systems are typically part of more than one SoS, and the involvement can vary over time, e.g., machines are added, removed or updated with new features.

A typical hazardous scenario could be that constituent systems may change state due to internal reasons and possibly assume a change in operation that another constituent system do not. Providing a machine position, for example, may not

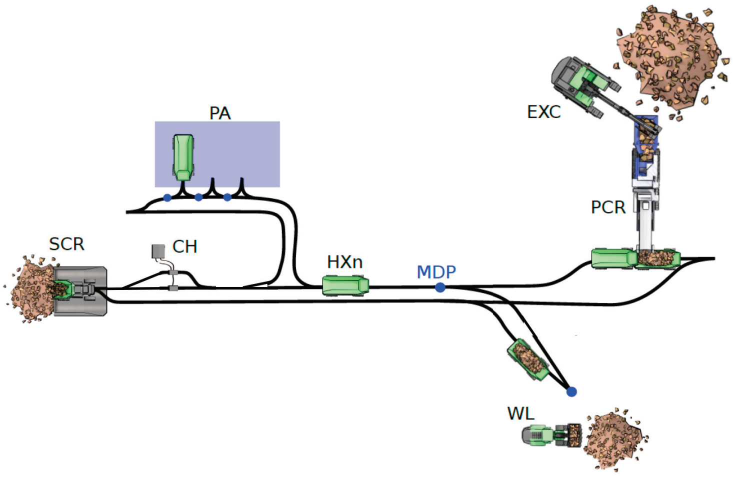

Fig. 1. An automated Quarry

be considered when the machine is in repair mode. Another typical critical scenario could be that a certain system relies on the correctness of information that is shared by another. A critical situation can occur, if signals are provided incorrectly, or interpreted differently by the receiver. Such hazards would not show up when analyzing hazards of the single system by itself.

III. Hazard Analysis Methods

There are mature hazard and safety analysis methods in literature, which are applied in industry today. Among the most well-known methods are Fault Tree Analysis (FTA), Failure Mode and Effect Analysis (FMEA) and Hazard and Operability Analysis (HAZOP). FTA is a top-down technique where each possible unwanted state is investigated based on which combinations of events could lead up to it. FMEA is a bottom-up approach where foreseeable faults of components of a system are analyzed with respect to likelihood and negative effects. Hazard and Operability analysis (Hazop), has its roots in the chemical industry and is utilizing guide words for identifying hazards and critical scenarios with respect to operations.

We have especially looked at the systems theoretic process analysis (STPA) [5], which is a method to model accident causation.

IV. Requirements on a low-footprint hazard analysis method

In our work we have so far applied FMEA and STPA, but run into problems with completing due to complexity and unfeasible work effort. Thus we strive for a method with lower footprint that would still aid in analyzing the potentially hazardous interactions within our quarry. We aim to tailor a method that provides:

- Abstraction of each system detail just the interaction and collaboration between constituents in the SoS should be analyzed. This includes state changes such as start-up, and maintenance breaks, but not internal handling of them.
- Reasonable footprint - the system must be described in such a way that complexity is manageable from a work effort perspective.
- Effective in finding hazards. In order to be meaningful, the method should find hazards that are not apparent at a first glance.

V. Analysis

The STPA method includes defining a controls structure that encompass which entities control which and what control signals that are involved. After the control structure is defined, the method is used to find possible loss scenarios. When we applied the STPA method, we saw a number of areas that presented challenges to us:

- The complexity of the system on the quarry did lead to high efforts for conducting STPA. Using all the items in the system blueprints that were given to us by engineers lead to an overly complicated control structure.
- It is important to describe or model the usage of a SoS. Not only the technical structure. There are control signals that are not shown in a technical schematic, e.g., a wave of hand by a manager.
- Analyzing many interacting smart products can cause a state explosion.
- Non-persistent analysis because products receive functional updates and thereby change behavior. There are rarely defined limits as of how much a product behavior can change when its software is updated.
- Hazards can be caused by simultaneous changes in control signals. We see that such hazards are difficult to identify in complex SoS.

VI. Proposed tailored method of STPA

One major finding from applying STPA in our case is that it is difficult to find the right level of detail for a control structure. Too much detail leads to a situation with too many signals (control actions) which in turn lead to high effort for performing the analysis. A second finding is that it is very easy to focus too much on system internals when analyzing the loss scenarios. Instead, we propose to focus on only the interaction between systems in the SoS, in order to avoid getting stuck in details of a specific system. In order to come up with a light weight method, we have devised three principles to aid us in getting a handle on the high complexity of the system.

- As a first step in the "Define purpose" phase of STPA, we define only the constituent systems. No internals or internal control actions are revealed. We define the control structure based on this simplified model. This means that each constituent system can never be modelled with more than one box in the control diagram.
- We add a step where we define system usage for each constituent in the form of use case descriptions. Based on the use cases, we elicit all signals that are involved, and we perform the "unsafe control actions" analysis based on these signals. The STPA does not explicitly address use case description, and we advocate it as an intermediate step to aid in getting the control diagram right. By using the use-case descriptions we see a way to focus on only the signals that matter rather than going through all signals that exist between systems.
- We perform an extra step of checking the signals for simultaneous changes that could cause hazards.

VII. Application in Case

By applying our method we got the control structure diagram described in Fig 2.

We go through the usage for each actor and define use cases. Based on our use cases we filter out each safety critical control signal and use that as an input to analysis of unsafe control actions. When the signals are listed in a table, we also check for problems caused by simultaneous changes. We did see indications of potential problems in scenarios when two different actors try to simultaneously change state of the same system.

VIII. Conclusion

Performing a hazard analysis is an important task when designing a complex directed SoS and many safety methods are aimed at single systems. We have applied STPA in an industrial case of a quarry and elaborated on our approach. When faced with the drawings and complex description of an industrial system there is a need to simplify and leverage the analysis procedure. We have come up with three principles to tailor the STPA procedure. We present the case and an example of the simplified control model.

References

[1] C. Ericson, *Hazard analysis techniques for system safety*. Wileys, 2015.

[2] Volvo Construction Equipment, "Electric Site Project." [Online]. Available: https://www.volvoce.com/global/en/news-and-events/news-and-press-releases/2017/conexpo-vegas-2017/volvo-ce-unveils-the-next-generation-of-its-electric-load-carrier-concept/

[3] A. Jafari, J. J. S. Nair, S. Baumgart, and M. Sirjani, "Safe and Efficient Fleet Operation for Autonomous Machines: An Actor-based Approach," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, ser. SAC '18, 2018.

[4] M. W. Maier, "Architecting Principles for Systems-of-Systems," *INCOSE International Symposium*, vol. 6, no. 1, pp. 565–573, 1996.

[5] N. G. Leveson and J. P. Thomas, *STPA Handbook*, 2018.

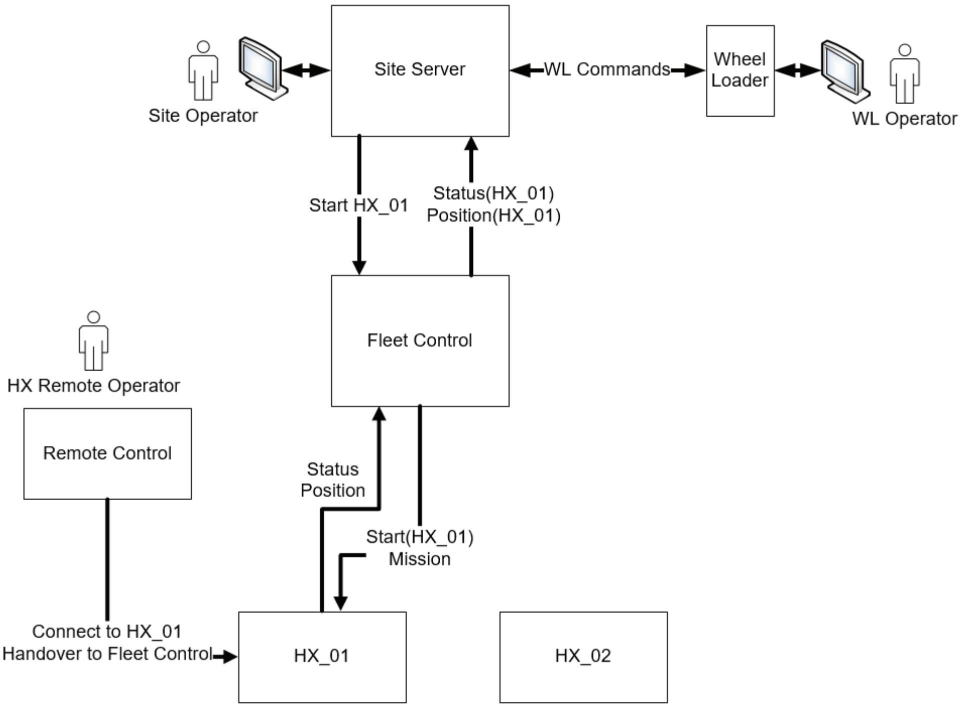

Fig. 2. Control Structure Diagram for STPA to study concepts.

Through our international collaboration programmes with academia, industry, and the public sector, we ensure the competitiveness of the Swedish business community on an international level and contribute to a sustainable society. Our 2,200 employees support and promote all manner of innovative processes, and our roughly 100 testbeds and demonstration facilities are instrumental in developing the future-proofing of products, technologies, and services. RISE Research Institutes of Sweden is fully owned by the Swedish state.

I internationell samverkan med akademi, näringsliv och offentlig sektor bidrar vi till ett konkurrenskraftigt näringsliv och ett hållbart samhälle. RISE 2 200 medarbetare driver och stöder alla typer av innovationsprocesser. Vi erbjuder ett 100-tal test- och demonstrationsmiljöer för framtidssäkra produkter, tekniker och tjänster. RISE Research Institutes of Sweden ägs av svenska staten.

RISE Research Institutes of Sweden AB
Box 1263, SE-164 29 Kista, Sweden
Telephone: +46 10 516 50 00
E-mail: info@ri.se, Internet: www.ri.se

SICS
RISE Report 2018:69
ISBN: 978-91-88907-18-9