# Towards Safe and Secure Systems of Systems

## Challenges and Opportunities

Avenir Kobetski
SICS Swedish ICT AB
Kista, Sweden
avenir.kobetski@sics.se

Jakob Axelsson
SICS Swedish ICT AB
Kista, Sweden
jakob.axelsson@sics.se

## ABSTRACT

While systems of systems (SoS) are starting to reach the market, it is not entirely evident how to analyze safety, and on a high level also security, of such systems. In fact, specific characteristics of SoS, such as independence, changing constitution, evolutionary development, and emergent behavior, provide certain challenges to the safety analysis. In this paper, such challenges are summarized and a systems theoretic safety analysis method, abbreviated as STAMP, is evaluated on an automotive SoS application example. In conclusion, STAMP seems well positioned to serve as a base for a future method for safety and, to a certain degree, security analysis of SoS, although some work remains to be done. The advantages and limitations of the STAMP approach when dealing with SoS are discussed.

## CCS Concepts

**•Security and privacy → Distributed systems security; •Software and its engineering → Software safety; •Computer systems organization →** *Embedded and cyber-physical systems;*

## Keywords

Systems of systems; safety; security; automotive; platooning

## 1. INTRODUCTION

The term *systems of systems (SoS)* started to become relevant some 20 years ago, and accelerated as a research area during the last decade. With the decreasing cost of communication technology, bringing along a surging number of applications of connected devices, SoS is no longer a far-fetched dream, but rather a reality within a wide variety of application domains.

Lately, different SoS ideas have been gaining popularity in the automotive industry, ranging from route planning to intelligent intersection management systems, etc. A good example of SoS is vehicle platooning, where a lead vehicle is followed by other vehicles that are driven more or less autonomously at a very short distance between each other, and possibly coordinated by a higher level controller. The motivator for platooning is primarily to improve fuel consumption by reducing aerodynamic drag, which is good both for the economy of the truck operator and for the environment. However, due to automation and short distances between the vehicles, safety obviously becomes an issue [3].

The automotive industry has a long tradition in improving safety, and best practices have recently been standardized as ISO 26262, encouraging the vehicle manufacturers to ensure that their product are safe to use. However, while technological development leads the current evolution of the SoS field, non-functional properties, such as safety and security, are lagging behind [2]. In fact, when the product is to become a part of an SoS, carrying out the safety analysis only on the product is not sufficient. Instead, safety is an emergent property that has to be dealt with on the SoS level.

In this work, we took advantage of the maturity of the automotive field and a state of the art safety analysis method, to study how an example platooning SoS application could be analyzed for safety and security. The outcome is a set of questions and challenges, but also opportunities, that the SoS community is facing when it comes to safety and security analysis of SoS.

Section 2 outlines typical SoS characteristics and the effects they have on challenges to safety analysis. In Section 3, a method for analyzing safety (and security), based on systems thinking is reviewed. Section 4 presents excerpts from an application of this method to a platooning example, while Section 5 concludes the paper.

## 2. SAFETY CHALLENGES RELATED TO SOS CHARACTERISTICS

While SoS definitions differ, there is a growing consensus about the distinguishing SoS characteristics, see e.g. [2, 6]. This section reviews such characteristics and discusses their implications on the challenges to safety analysis.

**Operational and managerial independence** means that the constituent systems not only can operate independently in a meaningful way, but also do so, even while being part

*SAC'17, April 3-7, 2017, Marrakesh, Morocco*
Copyright 2017 ACM 978-1-4503-4486-9/17/04...$15.00
http://dx.doi.org/xx.xxxx/xxxxxxx.xxxxxxx

of the SoS. They are developed, maintained, acquired, and owned separately. This means that all safety requirements on the SoS level have to be agreed upon between the different stakeholders. In this process, the requirements must be balanced with the individual safety and functionality requirements of each constituent system. Further, the implementation of safety requirement needs to be verified with respect to a common interface. How that should be done and by whom are often open questions.

**Evolutionary development** states that SoS do not appear fully formed, and functions and purposes may be altered based on experience. Also, the functionality of the constituent systems and even the composition of an SoS may change over time, perhaps more rapidly and more unexpectedly than in a typical system. Thus, there is a need for flexible and traceable safety analysis methods, that are able to monitor SoS evolution and adapt to it.

The main purposes of the SoS are fulfilled by behaviors that cannot be localized to any individual constituent system, but are rather attributed to the **emergent behavior**. The ability to foresee and analyze such behavior is a challenge per se. Appropriate safety analysis methods should at the very least be hierarchical and able to capture important safety risks at different levels of the SoS design.

Also, the constituent systems in an SoS are typically **geographically distributed.** This means that they have to rely on communication links, which poses additional challenges to both safety and security of the SoS.

In addition, while not unique to SoS, there are several system characteristics that are more pronounced in the SoS case.

**Complex interactions** between the constituent systems is one such characteristic. This is challenging in the safety analysis of any system. The independence properties add to the complexity through uncertainty about how the constituent systems would prioritize between their own goals and the goals of the SoS.

Also, SoS often exhibit a so-called **socio-technical composition**, involving both humans and machines that interact in complex ways. High complexity of SoS makes it more challenging for humans to correctly operate an SoS, especially in semi-automated systems. The human factor should thus be analyzed thoroughly. In addition, while in stand-alone systems human mistakes may be prevented using appropriate training, leadership, processes, etc., this assumes that there is someone in charge, both of the personnel and safety issues, which may not always be the case for SoS.

**Partial design** is a consequence of the evolutionary nature of SoS. In many cases, future design of an SoS may not be fully known at the time of safety analysis. The SoS safety analysis needs to somehow handle this uncertainty, e.g. unspecified technology, unspecified allocation of functionality, unspecified architecture, etc. This situation resembles early stages of a product development, when the design is not fully documented but is rather in the minds of the engineers. However, since an SoS may be developed independently by several actors, it might become quite challenging to transform informal knowledge about each individual system into safety requirements for the whole SoS.

**Other trade-offs** - SoS-level safety measures may impose on the constituent systems to rely on other systems for their safety, while providing certain information about themselves in order to do so. This raises questions of trust, privacy of conveyed information, as well as security. Trade-offs between safety and security will often be necessary and should thus be considered early in the development of an SoS.

In summary, to address SoS-related challenges, the safety analysis method should consider system hierarchy, emergence, uncertainty, interactions and trade-offs, both between systems and their internal goals, and the human factor. A system-based approach to safety analysis seems necessary.

## 3. SYSTEMS-THEORETIC ACCIDENT MODEL AND PROCESSES

*Systems-Theoretic Accident Model and Processes (STAMP)* is a systems theoretic approach to safety analysis [5]. In contrast to traditional accident causality models, such as Failure Mode and Effect Analysis (FMEA) [7], the focus is shifted from chains of failure events to a systemic view of possible undesired losses.

In STAMP, systems are treated as interrelated, dynamic processes that are continuously adapting to changing internal and external (environmental) conditions. Accidents are considered to be complex dynamic processes, resulting from flawed control mechanisms, involving interactions among people, societal and organizational structures, engineering activities, and physical system components, often on different hierarchical levels.

The STAMP approach includes a structured method for constructing a control model [4], starting from a less formal system design, e.g. concepts of operations (ConOps) [1].

## 4. APPLICATION OF STAMP TO SOS

In this section, STAMP is exemplified through excerpts from a case study that was carried out to gain a better understanding about which concepts are relevant for the safety analysis of SoS. The example application is a vehicle platoon, coordinated by a central route management system (RM). There is a driver (D1 and D2) in each vehicle (V1/V2), responsible for the lateral steering of the vehicle, and also able to override acceleration commands issued by the vehicle's platoon controller (PC1/PC2).

The first STAMP step is to identify the sets of *undesirable losses* (accidents) and system *hazards*, which are defined as system states that might lead to an accident in case of unfavorable conditions. Undesirable losses are typically defined rather generically and range from personal injuries to loss of property, and even loss of reputation. The set of hazards is often more specific for a given application and contains in our case incorrect separation distance between the vehicles (either too short or too long), incorrect lateral position, regulation violation, etc.

From hazards, a first set of *safety requirements* is deduced. Continuing on the above example, our requirements would include keeping the distance between platooning vehicles within certain boundaries (both taking advantage of platooning effects on fuel saving and being able to brake in time if necessary), keeping a prescribed lateral position, not violating any regulations (e.g. speeding or platooning in areas where such constructs are not allowed), and so on.

Next, a *control structure* needs to be identified, with the goal of keeping the SoS from entering hazardous states. Here, both a top-down and bottom-up approaches may be used, and eventually, it is advisable to exploit the hierarchical and modular nature of STAMP to analyze both high-level SoS behavior and the individual systems.

In this work, the focus is mainly on the SoS-level, and Figure 1 shows the high-level view of the constituent systems, with a special focus on the main control actions available to the different controllers. At the top of the control hierarchy, RM issues requests for formation and dissolution of platoons. However, it is unclear from the ConOps at hand how these requests are supposed to be processed. For example, should the driver always authorize a new formation, or can this be acknowledged automatically by the PC? This lack of design information, so typical to SoS, is represented by the dashed line in Figure 1. The driver selects PC's mode and steers the vehicle, while both D and PC have acceleration and brake commands at their disposal.

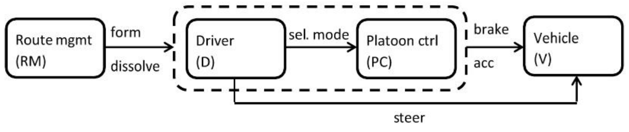

**Figure 1: A simplified example control model.**

With the control structure in place, the next STAMP step is to systematically go through all control actions and identify in which situations they may lead to any of the hazards identified previously. An example of the so called *unsafe control action (UCA)* is PC2 accelerating when inter-vehicle distance is already too short. A related UCA, which instead is caused by the lack of an appropriate action, is due to not braking in the same situation. On a higher level, RM not providing a dissolution request when platooning is no longer allowed (or beneficial for its participants) also results in an UCA. Note that in accordance with the STAMP approach, this UCA interprets safety in a broader way, including non-safety critical, yet undesired losses. In spite of apparent simplicity of the example, 34 UCAs were identified in total.

While UCAs already give a good picture of what may go wrong, and are transformed into additional safety requirements, they typically do not highlight the underlying causes to hazards. The next step is thus to dig deeper into the specifics of each UCA, and consider whether it could be caused by any part of the control model which encompasses that particular UCA. The *causal analysis* focuses in turn on the available sensors and actuators (and their potential malfunctioning), the controller itself and its internal model of the controlled process, any disturbances or conflicting control actions that the process may receive, as well as interactions between the controller and other, possibly superior, controllers, see [5] for more details. In the end, potential causes to UCAs are used to generate safety or security requirements, depending on the nature of the cause.

Returning to our example, we focus our presentation on the possible causes to RM not sending a dissolution request.

**Control input or external information wrong or missing.** One reason could be that RM has received information, potentially malicious, or even control input from a higher level controller, that platooning should always be allowed. Although there is no higher level controller in our control model, it is nonetheless reasonable to allow such possibility for two reasons. Firstly, RM is only vaguely defined in ConOps and its exact relationship to the outside world is a bit unclear. In fact, this seems to be a pattern that while system-level constructs are often well developed, higher levels of SoS are typically defined more loosely. And secondly, the structure of the platoon SoS is expected to evolve, and the safety analysis should take height for it. Thus, this analysis leaves a placeholder, to be filled in if an interface to a higher level controller is added to RM.

**Inadequate feedback / incorrect process model.** Another reason for RM being inactive is that process information is somehow incorrect. This could in turn depend on either RM's world map being wrong or outdated, or its estimation of the platoon position being wrong. Digging further, the estimation error could be either caused by accumulated incorrect position updates, or by an incorrect initialization value. In turn, incorrect updates may depend on failing sensor equipment, but also on shortcomings of the process model. For example, GPS typically has a slight measurement uncertainty. In the best case, this could be adjusted for at RM by combining information from various sources. However, under less favorable conditions, this could possibly lead to error drift.

**Inadequate control algorithm or control action.** Further, RM may send incorrect control values due to an incorrect algorithm (e.g. if only formation requests are implemented), an incorrect communication interface (e.g. vehicles do not understand that a dissolution signal has been sent), or even security issues (e.g. the dissolution signal is intercepted or manipulated).

**Delayed operation.** It can also happen that the signal from RM arrives too late. This could be due to delayed feedback from the vehicle, or it could be caused by communication issues, and again security (e.g. denial of service attack) should be taken into consideration. Finally, the recipient side could experience delays, e.g. due to high processor load. Which system represents the recipient side, whether it is the driver, the PC, or even the vehicle itself, is unclear.

**Conflicting control actions.** When it comes to conflicting control actions, they can only be speculated about at this point, since the control model is so simplistic. Yet, this situation will to some degree be quite common in SoS applications, and it is important to allow a certain level of speculation, trying to cover a variety of possible SoS designs at an early stage.

An imaginable scenario is that RM's are geographically delimited, that is RM1 may be handing over the control to RM2 when the platoon passes a certain boundary, e.g. a country or a state border. A possible hazard source would then be inconsistencies in the hand-over. For example, how will the systems react if RM1 sends a dissolution request, while RM2 does not? What if the dissolution signal from RM1 has been delayed so much that the vehicles are already under RM2's jurisdiction, which in turn believes that RM1 has already carried out its control task? Of course, these issues are solvable, yet it is important to identify such gray zones of control and address them properly.

**Inadequate communication with another controller.** Another possible conflict scenario is the one between RM and the driver. What happens if RM sends a dissolution request, which is not heeded by the driver? Should RM maintain its responsibility over the platoon in such a case? Is this even technically possible to disobey? The answer depends on how the systems within the dashed area in Figure 1 are implemented. Yet, regardless of the design, these questions should be raised and, ultimately, answered.

One could also double-check the validity of the above reasoning by reflecting about whether the scenario is realistic in the first place. Would a driver ever wish to disagree with RM? That may depend on the driver's own goals and how they relate to the scenario at hand. For example, the wish to save fuel, neglecting potential risks, is not that unlikely. While human behavior is often neglected in classical safety analysis, at least prior to a failure, STAMP takes the opposite stance. Indeed, since humans typically have a strong influence on both the operational behavior and technical design of systems, it makes sense to consider them as important controller components in the overall system model. We argue that this might be even more important in the SoS case, since SoS will typically be more complex and any deviation from normal operation should be detected and communicated to the participating systems.

## 5. CONCLUSIONS AND FUTURE WORK

The work presented here started with the question of how to conduct safety analysis of SoS. Typical SoS characteristics, and associated challenges to safety analysis, led us to believe that a systems based approach is imperative and STAMP was chosen as a promising method for assessment. STAMP was evaluated on an example application of automotive SoS, consisting of vehicles moving in platoon formation, their drivers, and supporting systems.

This paper summarizes generic challenges to the safety analysis that are typical for SoS. Next, it presents the STAMP method and discusses how STAMP could be applied to SoS, exemplified by extracts from an analysis of an automotive SoS application example.

While answering some questions, this paper raises even more. While STAMP seems as a promising method for safety analysis of complex systems, it is in need of further development to be applicable to SoS. For example, there is a need for a structured method for reasoning about trade-offs between safety requirements of different systems in an SoS, as well as between system-level and SoS goals. Also, SoS often experience a high level of uncertainty, both about how other constituent systems may operate, but also about how SoS functionality, and available interfaces, will evolve. The independence of the constituent systems makes the challenge even harder. One way forward is to provide several possible scenarios as a result of safety analysis, possibly enhanced with placeholders for evolving functionality.

On the other hand, the uncertainty inherent to SoS also leads to flexibility and increased design possibilities, e.g. allocation of safety functionality. How such allocation should be done, and how to monitor and reallocate functionality in case the SoS constitution suddenly changes (e.g. a vehicle leaves the platoon) are open questions.

As a side effect of this work, we realized that safety analysis should not be separated from security issues. In fact, using STAMP to analyze security risks would often follow exactly the same steps, through the list of hazards and unsafe control actions, up to and including parts of the causal analysis. The difference will typically lie in the final technical cause for a hazard, e.g. delayed communication due to an attack (security) instead of a software bug (safety), which motivates the use of the same analysis for both safety and security.

Also, solutions to safety and security risks may often end up in conflict with each other, e.g. a strongly encrypted communication may lead to longer delays, with adversary effects on safety. This makes it even more important to consider safety and security simultaneously, and make appropriate trade-offs between them and the functional goals of the individual systems, and the SoS as a whole.

Unification and solution of the above challenges into one coherent framework for safety analysis of SoS lies ahead and needs certain theoretical development. However, STAMP seems well positioned to serve as a basis in that direction.

## 6. ACKNOWLEDGMENTS

This work was supported by The Swedish Innovation Agency (Vinnova) through grant 2015-04840.

## 7. REFERENCES

[1] IEEE guide for information technology - system definition - concept of operations (conops) document. *IEEE Std. 1362-1998*, 2007.

[2] J. Axelsson. Systems-of-systems for border-crossing innovation in the digitized society: A strategic research and innovation agenda for sweden. 2015.

[3] J. Axelsson. Safety in vehicle platooning: A systematic literature review. *IEEE Transactions on Intelligent Transportation Systems*, 2016. to appear.

[4] C. H. Fleming. *Safety-driven Early Concept Analysis and Development.* PhD thesis, MIT, 2015.

[5] N. Leveson. *Engineering a safer world: Systems thinking applied to safety.* Mit Press, 2011.

[6] M. W. Maier. Architecting principles for systems-of-systems. In *INCOSE International Symposium*, volume 6, pages 565–573, 1996.

[7] D. H. Stamatis. *Failure mode and effect analysis: FMEA from theory to execution.* ASQ Quality Press, 2003.