Towards a Risk Analysis Method for Systems-of-Systems Based on Systems Thinking

Jakob Axelsson
Software and Systems Engineering Laboratory
Swedish Institute of Computer Science (RISE SICS)
Kista, Sweden
jakob.axelsson@ri.se

Avenir Kobetski
Software and Systems Engineering Laboratory
Swedish Institute of Computer Science (RISE SICS)
Kista, Sweden
avenir.kobetski@ri.se

***Abstract*— The characteristics of systems-of-systems (SoS) present fundamental challenges regarding properties such as safety, security, reliability, and robustness. This is due to the SoS nature where a collection of independent systems cooperate to fulfil certain high-level objectives. Risk analysis is thus an important activity in SoS engineering. This paper presents a risk analysis method which extends the existing STAMP safety analysis method that is based on systems thinking. Our extensions are aimed at coping with other risks than safety, and the usage is tailored to SoS. The method aims at deriving requirements on the constituent systems that will reduce the emergent risks on the SoS as a whole. The method has been applied to a case study of vehicle platooning.**

Keywords—systems-of-systems; risk analysis; safety; platooning.

I. INTRODUCTION

Systems-of-systems (SoS), with their origins in primarily the defense sector, are now rapidly becoming increasingly relevant in a large number of commercial applications as a result of the software-driven digitalization and automation of industry and society. Examples exist in domains such as transportation, energy, health care, manufacturing, smart cities, etc. Very often, the applications include the control of physical devices, and this makes them critical from several perspectives. This includes safety, since the physical devices may cause harm to humans, but also security, since they will handle information whose exposure could imply significant loss of value. Further, the fulfillment of the SoS mission is critical, and reliability and robustness to changing circumstances are key aspects. This is also reflected in the SoS scientific literature, where risk management is an important topic and includes properties such as sustainability, effectiveness, efficiency, safety, security, and reliability [1].

For this reason, risk analysis becomes essential in the development of SoS, and the particular SoS characteristics, where the SoS can be very long lived, and where each constituent system (CS) evolves over time, make it necessary to have a broad and life-cycle oriented perspective on risk analysis. Often, the different properties already mentioned become interrelated in the context of an SoS, as exemplified by the fact that a security vulnerability that allows someone to tamper with physical devices in the SoS can also lead to safety risks. Therefore, it makes sense to have a general risk analysis method for SoS, rather than specific methods for different properties. A generic method also allows SoS engineers to trade off different risks against each other within the same framework.

In this paper, we will present results from our ongoing research on *SoS risk analysis methods*. As will be explained in the paper, there are compelling reasons why systems thinking is a reasonable foundation for such an analysis, and we have therefore based our method primarily on the existing safety analysis method STAMP (Systems-Theoretic Accident Model and Processes) [2].

Our contribution is to generalize this method towards other risks, and to specialize it to the characteristics of SoS, in order to make it effective and efficient within this domain. The focus of our version of the method is the SoS level, whereas each individual CS will need their own risk analysis. The outcome is therefore primarily requirements on the CS's and their interfaces. Those requirements should be fulfilled in order to minimize the identified risk of SoS operations. At the same time, the requirements should not make too many assumptions about the implementation details of each CS.

A lot of the literature on SoS covers ultra-large, societal scale systems [1], but there is nothing in the definition of an SoS that excludes also smaller constellations. In fact, with the advent of technologies such as Internet of Things, SoS will appear increasingly in commercial products, and a risk analysis method should be applicable also in this context. We are interested in an approach that can support such applications, and also more flexible ways of creating SoS using e.g. software plug-in mechanisms [3].

The remainder of the paper is structured as follows: In the next section, the concept of risk is described in more detail, including how it relates to SoS. In Section 3, systems thinking is introduced, and is used to describe a generic model of an SoS. In Section 4, the proposed SoS risk analysis method is presented, and Section 5 illustrates its use in a truck platooning example. In Section 6, some related work is described, and in the final section, the conclusions are summarized together with some directions for future research.

II. RISKS IN SYSTEMS-OF-SYSTEMS

In this section, a more precise definition of risk will be given. It is then discussed how the special SoS characteristics

This research has in part been supported by Vinnova through grants 2015-04881 and 2016-04232, and by the Volvo Group.

relate to risk, and some delimitations are made regarding the scope of this paper.

A. Definition of Risk

A fairly recent definition of risk has been provided by the standard ISO31000 [4], which uses the term to mean the "effect of uncertainty on objectives". This definition is however not consistent with the everyday usage of the word, which normally associates risks with negative values, whereas the standard also allows positive effects to be regarded as risks (something that would usually be called opportunities instead). In this work, the aim is to reduce negative effects, and hence we will use the following more restrictive definition:

Definition. *Risk is the negative effect of uncertainty on objectives.*

A "negative effect on objectives" would typically be a loss of value to some stakeholder, and "uncertainty" can be seen as the occurrence of events over which the system does not have full control. An event is typically that a particular system state is entered, and that the environment is in a certain condition, and it is the combination of these two that results in the loss. A typical risk can thus be formulated as follows:

<Loss of value occurs> **if** <System is in hazardous state> **when** <Environmental condition applies>.

As an example, a traffic related risk could be "Pedestrian gets injured" **if** "Car brakes do not work" **when** "Pedestrian walks into road in front of car". Both the system being in the hazardous state and the environmental conditions are uncertain events which may or may not apply, and the loss of value is thus an effect of uncertainty. But it is the combination of the system state and environmental condition that leads to the loss, and not just one of them. Risk analysis and mitigation is mostly about identifying the elements of such risk expressions, and trying to eliminate them, or at least to reduce their probability of occurrence or their severity.

B. Relating Risk to SoS Characteristics

The foundation for our method is the key characteristics of SoS [5], which lead to certain needs when it comes to risk analysis:

- The *operational independence* puts limitations on what risk mitigation techniques are feasible, but also provides opportunities if different CS's provide redundancy.
- The *managerial independence* means that risk management must be a collaboration between the organizations behind the CS's and the SoS. This means that not only technical but also organizational elements must be included; the analysis should be socio-technical.
- The *evolutionary development* makes it necessary to have a continuous approach to risk analysis and management.
- The *emergent behavior* entails that there is a need for a hierarchical view on the SoS.
- The *geographical distribution* implies that the interactions are primarily information oriented, with key parts being implemented in software, and the method thus needs to capture software characteristics well.

The general properties of SoS are thus more restrictive than for systems in general, and our hypothesis is that it makes sense to have a tailored method to make the risk analysis efficient. This is particularly important since the evolution of SoS means that parts of the risk analysis will have to be redone repeatedly.

C. Risk Analysis Scope

The system-of-interest for the risk analysis in this paper is the SoS, and risks related to using that system are thus in focus. The definition of risk presented in Section 2.1 is very broad, and covers any type of negative effects to a stakeholder. In this paper, the focus is narrower, and delimited primarily to operational risks, i.e. negative effects that are caused by the system during its use. This includes risks related to safety, security, reliability, etc., which are consequences of the *function* of the SoS.

Other categories of risk, such as development cost or schedule, are primarily relate to another system-of-interest, namely the development system which consists of the people and tools that carry out the task of designing and implementing the SoS [6]. During that process, the SoS is treated as an entity without any behavior that is handled by functions in the development system.

The fact that the operational phase is in focus does not mean that other life-cycle phases are ignored. Risk mitigation actions can be assigned to any life-cycle phase, and the risk analysis method as such is primarily executed during the development phase.

III. SYSTEMS THINKING AND SOS

The generic SoS characteristics are pointing in the direction of systems thinking as a theoretical foundation for SoS risk analysis, since it offers whole-part hierarchy, information flows between functions including feedback loops, and the ability to deal with many kinds of elements, including humans and software. In this section, we will introduce the principles of systems thinking, discuss its use as a theoretical framework, and show how a generic SoS can be modelled using the language of systems thinking.

A. Principles of Systems Thinking

Systems thinking is often seen as the theoretical foundation on which rests the more practically oriented systems engineering discipline. In systems thinking, a key relation is between the *system* (the whole) and its *elements* or parts, where the elements are interacting with each other. The relation between the system and its elements is recursive, so an element of one system may be viewed as a system in itself, with its own elements, thereby creating a *hierarchy*. By composing the elements in a certain way, properties and behavior are created which cannot be attributed to any of the individual elements in isolation, and must hence be regarded as properties and

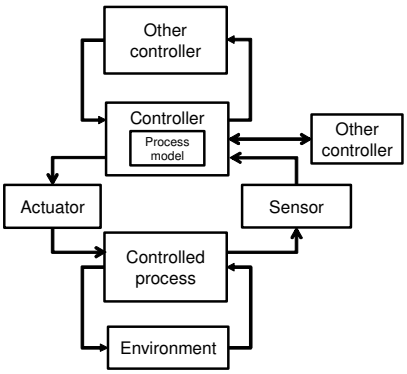

Fig. 1. Control diagram concepts for modeling a system.

behavior of the system. This is referred to as *emergent* properties and behavior. The fundamental idea in systems thinking is that the system cannot be analyzed by looking just at the individual elements, but must be seen as a whole, to capture this emergence. Element interactions are important in this, and especially various kinds of *feedback loops*, both negative (stabilizing) and positive (amplifying) ones. These interactions are central when designing a system to achieve a certain desired emergent behavior, and to avoid undesired behavior [7].

A distinction is sometimes made between *hard systems* and *soft systems*, where hard systems are typically dominated by technical questions, and soft systems have a focus on humans and organizational aspects [8]. The latter typically uses methods which are less quantitative, and requires understanding the individuals' motivations and viewpoints.

It is common to model a system through a control diagram (see Fig. 1), where the boxes are functional elements, such as controllers, controlled processes, sensors, and actuators. The arrows represent control flows, i.e. information transfers, between the functional elements. Controllers can be hierarchical in any number of levels, and communicate with other controllers on the same level. A controller internally also maintains some model of its controlled process which it uses to evaluate different courses of action. Note that even though we have chosen a certain notation in the figure, standard modeling notations could equally well be used, including SysML models and control engineering models.

B. Theoretical Framework

Systems thinking has previously been the basis for Rasmussen's work focusing on safety analysis [9], which has later been refined by Leveson and packaged in the STAMP method [2]. The latter work takes a broad view on safety, including basically any hazards that can lead to some loss of value, and it has been applied not only to safety but also to security [10]. This is particularly important given the information centric nature of SoS, where a security problem can easily be turned into a safety issue. A version of STAMP, called the Systems Theoretic Early Concept Analysis (STECA), has also been developed to support the early phases of systems engineering [11].

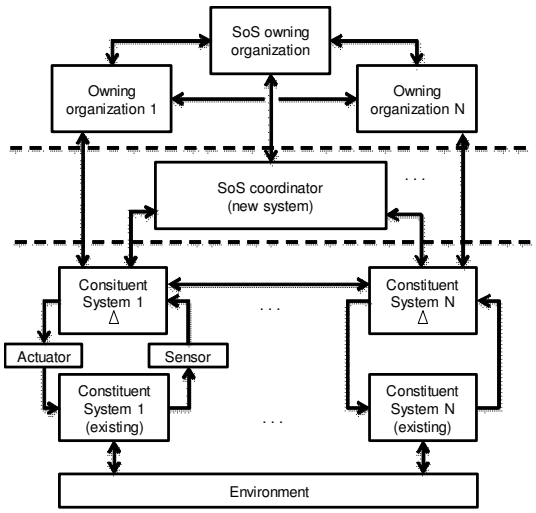

Fig. 2. Control diagram of a generic SoS model, showing the operational, coordinating, and managerial layers.

The basis for STAMP is a model of the system as a control diagram, based on the template in Fig. 1. Hazards in the system are associated with control actions, and to analyze what causes a hazard, each controller is investigated to see what could cause it to submit an inadequate control, and in what ways an adequate control action could not be followed.

C. A Generic SoS Control Model

When the system-of-interest is an SoS, there are certain elements that will appear in a control diagram, and this is illustrated generically in Fig. 2. The figure shows three hierarchical levels (although a concrete instantiation would likely subdivide some of the levels in several layers).

At the bottom are the constituent systems, that should co-operate to form the SoS. At least some of these are typically existing, but need to be adapted to the context of the SoS. Defining this adaptation is a key activity in the SoS engineering process, and in the figure the result has been illustrated as a CS Δ, which contains the added or changed functionality. The CS Δ is sometimes an add-on hardware that communicates with the underlying CS using sensors and actuators, and sometimes it is software that communicates directly with software in the existing CS. The CS Δ also allows the CS's to communicate with each other.

The middle level is for SoS coordination, which includes elements that are added specifically to make the SoS work. Typically, coordinator elements are needed that communicate with the CS Δ 's.

The highest level is organizational, and consists of the organizations behind each of the CS's, as well as the organization that is responsible for the SoS. As can be expected from Conway's law [12], this organizational level thus reflects the structure of the technical level.

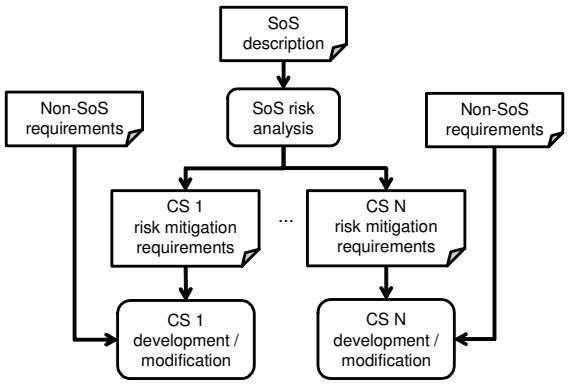

Fig. 3. Overview of SoS risk analysis context, showing how the analysis transforms an SoS description into CS requirements.

In the figure, the CS's are depicted as controlled processes, and the other elements are controllers.

IV. AN SOS RISK ANALYSIS METHOD

The SoS risk analysis method we are developing takes STAMP [2] as the starting point, since its characteristics appear to match the needs and properties of SoS. We have in particular found STECA [11] to be a particularly good starting point for an SoS analysis, since the description of the SoS is by its nature on a high level, reminiscent of the early phases of traditional systems engineering. However, the top-down approach in STECA should be complemented with a bottom-up view integrating the CS's, since SoS engineering to a large extend deals with making already existing CS's fit together to produce the desired emergent behavior.

Our work has primarily focused on adapting STAMP and STECA to the characteristics of an SoS, and generalizing it from safety to other risks, and it is also in this adaptation our contribution lies. It provides a tailoring specific to SoS engineering, instead of a completely generic method, which should considerably reduce the effort of its application.

The intention is that the method should be applied as an integral part of the SoS systems engineering effort. This gives certain constraints to the method, in particular the limited access to information about the CS's, and the integration centric, bottom-up way of working which differs from traditional systems engineering approaches that are typically top-down. It also guides analysts in the identification of mitigation actions of relevance to an SoS.

A. *Process Context and Overview*

Fig. 3 shows the process context of the SoS risk analysis. Its input is a description of the SoS, typically on the level of a Concept of Operations (ConOps) document. The output is a set of requirements to each CS, which they should fulfil to mitigate the risks identified in the analysis. These requirements go into the development or modification processes for the CS's (that result in the CS Δ's), together with whatever other requirements that may apply as a consequence of their operational and managerial independence.

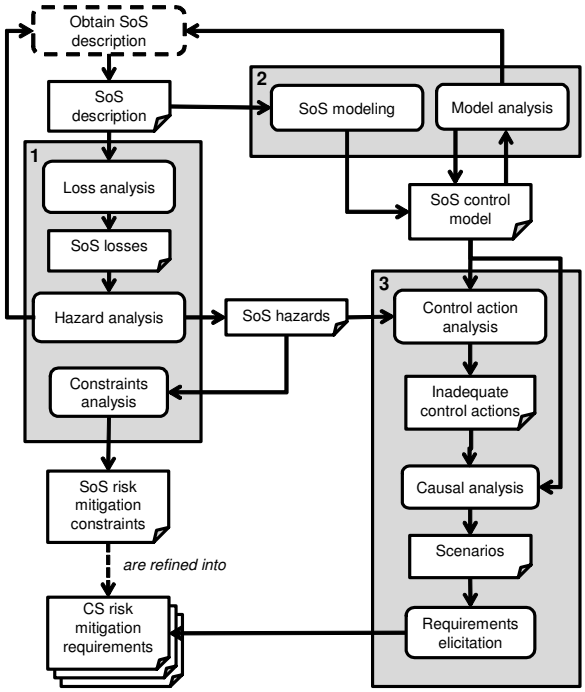

Fig. 4. Overview of SoS risk analysis process, with the main subprocesses (1) Loss & hazard analysis; (2) Modeling; and (3) Causal analysis.

An overview of the suggested risk analysis method is given in Fig. 4. It consists of three main steps, namely (1) Loss & hazard analysis; (2) Modeling; and (3) Causal analysis. Each of these steps contains further sub-activities.

As explained above, the input to the analysis is an SoS description. However, in practice this is hardly ever a given, and therefore the figure also illustrates that there needs to be a process which obtains the description. As a result of the analysis, it is also normal that missing information is discovered, which triggers feedback loops to look for more information and extend the SoS description.

We will now describe each of the activities in more detail.

B. *Loss and Hazard Analysis*

The Loss and Hazard Analysis activity has as its goal to identify and characterize the negative effects that can be caused by the SoS. We use the term "loss" for those effects (whereas STAMP uses the terms "accident" and "loss" more or less interchangeably). The activity consists of three sub-activities: Loss analysis; Hazard analysis; and Constraints analysis. They all deal with the SoS at a high level, based on the initial description.

1) Loss Analysis: In the loss analysis, the goal is to identify a set of high-level losses that the system should try to avoid, based on the SoS description. Normally, these should be quite few. By looking at a large number of examples, we have identified the following high-level categories that may be considered as a guideline in identifying adequate losses to categories of stakeholders:

- *Human death or injury*. The SoS causes harm to people, such as operators or bystanders.
- *Material damage*. The SoS causes damage to material.
- *Mission not fulfilled*. The SoS mission not fulfilled.
- *Information losses*. The SoS causes sensitive information to be exposed to non-authorized entities.
- *Economical damage*. This broad category covers things like loss of reputation, fines as a consequence of not meeting legal or contractual requirements, etc. (Of course, some of the previous categories may also lead to financial damage, but this category is primarily to capture those losses that are purely economic.)

In specific cases, other loss types may naturally be used, but this list is a good starting point in eliciting the losses. It is important to focus on SoS related losses. In other words, we are not interested in losses that occur also when a CS operates independently, but only those that are a consequence of being a part of the SoS.

The output from the activity is a numbered list of losses.

2) Hazard Analysis: Based on the losses from the previous step, hazards are identified. STAMP defines a hazard as "a system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident (loss)" [2]. It recommends to keep the list of hazards short, and not more than a dozen should be included, which indicates that they are on a high level.

The process for identifying hazards is basically to start with a loss, and consider (using domain knowledge) what SoS states or conditions that could lead to that loss.

The output from the activity is a numbered list of hazards, with references to the loss that they could incur. (Note that it is possible that one hazard could relate to several losses.) A possible outcome is also that the SoS description is modified in such a way that the hazard can be completely avoided, if this is a realistic option.

3) Constraints Analysis: Based on the hazards, a number of high-level constraints are defined. These are high-level requirements that the SoS should fulfil, and the subsequent analysis aims at breaking down these constraints in a structured way to concrete requirements that can be allocated to the CS's.

Often, there is a one-to-one mapping between hazards and constraints, where the constraint is the negation of the hazard. The output of the activity is a list of constraints, with relations to the corresponding hazards.

C. Modeling

The modeling activity aims at deriving a system-theoretic model of the SoS, that is suitable for risk analysis, similar to the generic structure in Fig. 2. It consists of two sub-activities: SoS modeling is the actual modeling activity, which is based on the SoS description; and Model analysis verifies the produced model for completeness, consistency, etc.

Modeling is often carried out in parallel to the Loss and hazard analysis. The modeling also results in the identification of missing or ambiguous information in the SoS description, which leads to updates in the document.

1) SoS Modeling: The modeling activity is basically a textual analysis of the SoS description document, as defined in STECA [11]. The analyst scans the text, and identifies concepts that should appear as elements (i.e., controllers, sensors, actuators, controlled processes, and process models) in the model.

Then, the elements are related to the appropriate hierarchical level, and the guidance here is that the higher the level, the longer the time horizon is, and the broader the geographical distribution. Higher levels control lower levels.

The information flows between the elements are identified based on the text in the SoS description and using domain knowledge. Note that domain knowledge is often partly proprietary to each CS owner, and the activity thus requires active involvement from them.

The output is an SoS control model using the structure of Fig. 1.

2) Model Analysis: The Model analysis activity takes the SoS control model and verifies it by looking for missing information, such as missing information flows in the control loops, and inconsistencies. STECA provides checklists for this.

The output of the activity is an updated SoS control model.

D. Causal analysis

The goal of the causal analysis is to find, starting from hazards and the SoS control model, how the system could cause losses to occur. Based on the issues identified, requirements are defined to prevent them from occurring. The activity is divided in three parts: Control action analysis; Causal analysis; and Requirements elicitation.

1) Control Action Analysis: The hazards could occur because some parts of the system do something inappropriate, and what the system does is identified by looking at all controllers in the model. The outputs of the controllers are called *control actions* and are what the system does. Therefore the analyst needs to go through each control action in the model, and check if it could in any way lead to any of the hazards.

When checking the control actions, STAMP provides a set of keywords to aid the analyst. These include that the control action is *not provided* when required; that it *is provided* when it is not expected; that it is *applied with wrong timing or order*; or that it is *stopped too soon or applied too long*.

Thus, for each applicable inadequate control action, we are looking for a formulation of the following kind:

If <control action> **is** <keyword> **when** <context> **then** <hazard>

Note that it is only the context that is added in this step, whereas the other elements are already given from the previous

steps. The context represents a state in the system, and the description is typically expressed in terms of the system's state variables.

2) Causal Analysis: The list of inadequate control actions is further analyzed to understand how they cause the hazard. For this, two steps are performed, that result in a set of scenarios describing causes for the hazards. The scenario is expressed by appending the formulation of the inadequate control action with "... **because** <cause>".

The first step tries to understand why the inadequate control action could occur, and this is checked by following the information flows backwards through the controller, and back to its inputs. This allows the discovery of reasons such as inadequate control algorithms; erroneous process models; missing or wrong control input; missing or inadequate feedback; sensor faults; delays, etc.

The second step focuses on the situation where an appropriate control action has been issued, but it is not followed. This requires the analyst to follow the information flows forward from the controller, and allows discovery of problems such as actuator faults; delays; conflicting control actions from other controllers; component failures in the controlled process; change over time, etc.

Note that the same cause may very well show up in several scenarios, and thus relate to several inadequate control actions (which relate to different hazards or different keywords).

E. Requirements Elicitation

The final step is to elicit requirements that remove the hazards from the system, or at least reduce their likelihood of occurring or their severity. This is done by looking at the scenarios from the causal analysis, and defining ways to prevent them from happening. This can either include adding a new controller or giving extended tasks to existing controllers.

The requirements need to be specific to a certain constituent system, or to the SoS controller, since these are the places where the behavior of the SoS is implemented.

V. CASE STUDY: TRUCK PLATOONING

As an initial validation, the SoS risk analysis method has been applied to a truck highway platooning application, with a focus on safety and mission risks. In this section, a few examples from that case study will be used to illustrate some of the analysis steps in the method.

A. Overview of the Platooning Application

The idea of highway platooning is that a lead vehicle, which is driven manually, is followed closely by a number of other vehicles using automated driving. When the concept was initially introduced, the focus was on improving road throughput to reduce congestions, but more recently, the emphasis has been on truck applications, where the aerodynamic drag can be substantially reduced by shortening the distance between the trucks, leading to reduced energy consumption. This is also the application area we have studied, and it constitutes a good example of an SoS, since the trucks can still drive independently, and they can only get the benefit of reduced fuel consumption by collaborating in a platoon.

Although it is obvious that safety is at risk when reducing inter-vehicle distances, platooning safety has not been extensively studied [13]. The safety analysis standard used by the industry [14] is very explicitly focused on an individual vehicle and on human safety, and therefore not sufficient for the platooning risk analysis, which was the motivation for developing a new method.

In the next subsection, the SoS modeling for this case is discussed. Then, in the following two subsections, the analysis of two different risks are described, by presenting the results from each of the analysis steps described in Section 4.

B. Modeling Platooning as an SoS

With reference to the generic SoS control structure in Fig. 2, the existing CS's are the current trucks, and CS Δ are the modifications needed to implement platooning functionality. There is also another controller in the lowest layer, namely the driver, which controls and monitors the truck and the CS Δ. Between the CS Δ's, there is a short-range radio link allowing them to exchange data, as well as sensors for measuring distances. On the SoS level, there is a platoon coordinator which assists trucks in finding each other on the highway, as well as restricting when platooning is allowed, depending on road conditions. On the organizational level, the owners and producers of the individual trucks can be found, as well as the organization that defines the interoperability standards that allow formation of platoons of different truck brands, and the organization that operates the platoon coordinator.

C. Example of a Safety Related Risk

The first example is a risk related to traditional safety, i.e. focusing on human injury:

1. *SoS losses*. The loss in focus in this analysis is "Platoon occupant gets injured".
2. *SoS hazards*. The loss could, under worst-case conditions, occur with the hazard "Too short separation distance".
3. *SoS risk mitigation constraints*. A high-level requirement on the SoS to avoid this hazard is "There shall always be a sufficient separation distance between two consecutive vehicles in the platoon to avoid collisions".
4. *Inadequate control actions*. One control action that causes this hazard is "**If** follower vehicle acceleration **is** provided **when** separation distance equals lower bound **then** separation distance becomes too short".
5. *Scenarios*. One reason why this acceleration is provided is "... **because** the distance sensor did not provide a correct measurement".
6. *CS risk mitigation requirements*. A requirement that mitigates this scenario is "Each vehicle shall always validate its primary distance measurements against a

secondary information source, and resume manual control if they are different".

D. Example of a Mission Related Risk

The second example illustrates a risk related to not fulfilling the mission of the SoS, which is in this case to reduce fuel consumption of the participating trucks:

1. *SoS losses*. The loss focused here is "Fuel reduction not achieved".
2. *SoS hazards*. The loss could, under worst-case conditions, occur with the hazard "Too long separation distance".
3. *SoS risk mitigation constraints*. A high-level requirement on the SoS to avoid this hazard is "There shall not be an excessive separation distance between two consecutive vehicles in the platoon".
4. *Inadequate control actions*. One control action that causes this hazard is "**If** follower vehicle acceleration **is** not provided **when** separation distance equals upper bound **then** separation distance becomes too long".
5. *Scenarios*. One reason this acceleration is not provided is "... **because** lead vehicle accelerates more than the follower is able to due to weight differences".
6. *CS risk mitigation requirements*. Two alternative (or complementary) mitigations to this are (a) "Before engaging in a platoon, the vehicles shall be placed in order of increasing acceleration ability"; (b) "Follower vehicle shall communicate if it is not able to accelerate sufficiently, in which case lead vehicle shall reduce its acceleration".

Note how the hazards of this and the previous examples are actually somewhat contradictory, which clearly indicates some of the challenges in building effective platooning in practice.

VI. Related Work

A number of authors have identified the importance to SoS of risk management, which is defined as to "Monitor, identify, assess, analyze, and mitigate risk encountered in the SoS" [15]. A partial taxonomy for different risks and conflicts in an SoS has been proposed [16], but it is rather broad and focuses on acquired SoS, which limits its use for commercial applications. Common SoS risks are discussed in [17], e.g. multiple stakeholders; multiple risk management processes; long life-cycles; technical risks; integration risks; functional performance risks; and interface complexity.

In [18], foundations of risk management for SoS are identified, which are undesirable consequences; uncertainty; and temporal domain. This leads to the identification of seven guiding questions for risk management: (a) What are the desirable events at a particular time? (b) What can go wrong? (c) What are the consequences? (d) What is the chance of occurrence? (e) What can we do to manage them? (f) What are the alternatives? (g) What are the effects beyond this particular time? A process is also proposed in [19], based on identified knowledge gaps in assessing risks, including complexity, ambiguity of consequences, and uncertainty about probabilities. It also suggests an analysis method with the following steps: (a) understand consequences; (b) identify hazards; (c) identify risk management strategies; (d) create a functional model of the system's response. A case study of a global maritime infrastructure with multiple stakeholders illustrates the analysis.

A systems perspective to SoS risk is common in much of the literature. In [20], a systemic approach to SoS risk management is presented, where a systemic risk originates from multiple sources, and is created by the interrelation between individual risks. The analysis is focused on program management related risks, rather than operational ones. Cross-cutting risks are also the focus in [21], which discusses a safety management system for disasters.

Much work on SoS risk management has a close relation to safety, and in [22], ten key challenges to SoS safety are derived and exemplified. A risk model for SoS, with a focus on safety, is proposed in [23]. It emphasizes the need to focus on SoS related risks, and not just any risk associated with a CS. The model is described as a fault tree, where four generic regions are identified which correspond to different levels in the tree. Region A treats generic top-level SoS loss classes; Region B describes concrete SoS losses of the different classes; Region C identifies hazards that could incur those losses and the conditions leading to a loss; and Region D looks at the states of the CS's that result in the hazards.

A number of analysis approaches to SoS risk have been suggested, and many of them focus on quantification on a very high level of abstraction. This includes [24], which uses the Inoperability Input-Output Model (IIM) to mathematically describe interdependent systems; the Hierarchical Holographic Modelling (HHM) to identify risk scenarios; and the Phantom System Model to provide justifications for investing in protecting against the risks. In [25], risks to a plant exposed to external events such as earthquakes are analyzed. The focus is on establishing probabilities of events, to support Monte Carlo simulations. The usage of portfolio optimization techniques to evaluate the value at risk is investigated in [26], and a model-based approach based on Bayesian Belief Networks with Monte Carlo simulations is suggested in [27].

The paper that comes closest to our approach is probably [28]. It proposes a methodology to support non-specialist end users in the identification, organization and discussion of information required to manage SoS evolution, and uses a modified form of HAZOPS (Hazard and Operability Study) to analyze the associated risks of evolution. It applies the method to a case study of an aircraft crash.

Some approaches focus on the dynamic nature of the SoS, such as [29] which suggests the use of precursor analysis (a.k.a. leading indicators [30]), which are early signs of system failure. The analysis is applied to large critical infrastructure. The need for dynamic risk assessments is also argued in [31]. The approach is not very specific to SoS, and seems to focus on limiting the risks in particular operational circumstances.

Research on SoS resilience can also provide information related to risk mitigation, such as [32] which identifies a

number of tactics to improve resilience, and [33] which argues the use of systems thinking to deal with complex risks and interdependencies.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented an initial version of a risk analysis method for SoS. The method is based on systems thinking, and heavily influenced by the existing safety analysis method STAMP. Our contribution has been to tailor the method for SoS, and generalize it somewhat to also cover other operational risks than safety. The method was illustrated using a case of truck platooning.

In general, our experience is that the approach works well to capture SoS risks. These risks can come from a broad range of types, which necessitates an analysis on an abstract level. One of the challenges in doing the analysis is to focus on what is actually within the SoS scope, and avoiding going down into risks specific to an CS that exist regardless if the CS is participating in the SoS or not. Assisting analysts in this is a strong motivation for having an SoS specific method, which provides guidance in scoping.

In the future, we intend to evolve this method further. This includes applying it in practice in more examples, which will result in a stronger knowledge base that can be used to provide better guidance for SoS analysis. Also, there is a need for some tool support, since the method involves a fair amount of book-keeping of data elements that are linked to each other in different ways. Extending the approach to also include quantitative analyses is potentially interesting, although the researchers behind STAMP discourage this due to the difficulty in providing accurate estimates of probabilities. A further extension would be to deal with the dynamic evolution of an SoS, and leading indicators could be a useful starting point.

REFERENCES

[1] J. Axelsson, "A Systematic Mapping of the Research Literature on System-of-Systems Engineering," in IEEE 10th Annual System of Systems Engineering Conference, 2015.

[2] N. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety. MIT Press, 2011.

[3] J. Axelsson and A. Kobetski, "On the conceptual design of a dynamic component model for reconfigurable AUTOSAR systems," in 5th Workshop on Adaptive and Reconfigurable Embedded Systems, 2013.

[4] ISO, "ISO 31000:2009, Risk management – Principles and guidelines," 2009.

[5] M. W. Maier, "Architecting Principles for Systems-of-Systems," INCOSE Int. Symp., vol. 6, no. 1, pp. 565–573, Jul. 1996.

[6] J. Axelsson, "Towards an Improved Understanding of Humans as the Components that Implement Systems Engineering," in INCOSE International Symposium, 2002, vol. 12, no. 1, pp. 1137–1142.

[7] R. L. Ackoff, "Towards a System of Systems Concepts," Manage. Sci., vol. 17, no. 11, pp. 661–671, Jul. 1971.

[8] P. Checkland, Systems thinking, systems practice. Chichester, England: John Wiley & Sons, 1993.

[9] J. Rasmussen, "Risk management in a dynamic society: A modelling problem," Safety Science, vol. 27, no. 2–3. Elsevier Sci B.V., pp. 183–213, 1997.

[10] W. Young and N. G. Leveson, "An integrated approach to safety and security based on systems theory," Commun. ACM, vol. 57, no. 2, pp. 31–35, Feb. 2014.

[11] C. H. Fleming and N. Leveson, "Integrating Systems Safety into Systems Engineering during Concept Development," in 25th Annual INCOSE International Symposium, 2015.

[12] M. E. Conway, "How do committees invent," Datamation, vol. 14, no. 4, pp. 28–31, 1968.

[13] J. Axelsson, "Safety in Vehicle Platooning: A Systematic Literature Review," IEEE Trans. Intell. Transp. Syst., pp. 1–13, 2016.

[14] ISO, "ISO 26262 Road Vehicles - Functional Safety," 2011.

[15] A. Gorod, B. Sauser, and J. Boardman, "System-of-Systems Engineering Management: A Review of Modern History and a Path Forward," IEEE Syst. J., vol. 2, no. 4, 2008.

[16] A. P. Sage, "Conflict and risk management in complex system of systems issues," in SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483), vol. 4, pp. 3296–3301.

[17] E. H. Conrow, "Risk Management for Systems of Systems," Crosstalk, pp. 8–12, 2005.

[18] C. A. Pinto, M. K. McShane, and I. Bozkurt, "System of systems perspective on risk: towards a unified concept," Int. J. Syst. Syst. Eng., vol. 3, no. 1, pp. 33–46, 2012.

[19] M. Bristow, L. Fang, and K. W. Hipel, "System of Systems Engineering and Risk Management of Extreme Events: Concepts and Case Study," Risk Anal., vol. 32, no. 11, pp. 1935–1955, Nov. 2012.

[20] S. J. Gandhi, A. Gorod, and B. Sauser, "A systemic approach to managing risks of SoS," in 2011 IEEE International Systems Conference, 2011, pp. 412–416.

[21] D. Prochazkova, "Identification and Management of Risks of System of Systems," Int. J. Comput. Inf. Technol., vol. 2, no. 2, pp. 2279–764, 2013.

[22] C. Harvey and N. A. Stanton, "Safety in System-of-Systems: Ten key challenges," Saf. Sci., vol. 70, pp. 358–366, 2014.

[23] J. M. Aitken, R. Alexander, and T. Kelly, "A risk modelling approach for a Communicating System of Systems," in 2011 IEEE International Systems Conference, 2011, pp. 442–447.

[24] Y. Y. Haimes, "Models for risk management of systems of systems," Int. J. Syst. Syst. Eng., vol. 12, no. 12, pp. 222–236, 2008.

[25] E. Zio and E. Ferrario, "A framework for the system-of-systems analysis of the risk for a safety-critical plant exposed to external events," Reliab. Eng. Syst. Saf., vol. 114, pp. 114–125, 2013.

[26] P. Shah, N. Davendralingam, and D. A. DeLaurentis, "A conditional value-at-risk approach to risk management in system-of-systems architectures," in 2015 10th System of Systems Engineering Conference (SoSE), 2015, pp. 457–462.

[27] A. Kinder, M. Henshaw, and C. Siemieniuch, "A model based approach to system of systems risk management," in 2015 10th System of Systems Engineering Conference (SoSE), 2015, pp. 122–127.

[28] R. Lock, "Developing a methodology to support the evolution of System of Systems using risk analysis," Syst. Eng., vol. 15, no. 1, pp. 62–73, Mar. 2012.

[29] Z. Guo and Y. Y. Haimes, "Risk Assessment of Infrastructure System of Systems with Precursor Analysis," Risk Anal., vol. 36, no. 8, pp. 1630–1643, Aug. 2016.

[30] N. Leveson, "A systems approach to risk management through leading safety indicators," Reliab. Eng. Syst. Saf., vol. 136, pp. 17–34, 2015.

[31] J. M. Aitken, R. Alexander, and T. Kelly, "A case for dynamic risk assessment in NEC systems of systems," in 2010 5th International Conference on System of Systems Engineering, 2010, pp. 1–6.

[32] P. Uday and K. Marais, "Designing Resilient Systems-of-Systems: A Survey of Metrics, Methods, and Challenges," Syst. Eng., vol. 18, no. 5, pp. 491–510, Oct. 2015.

[33] A. Cavallo and V. Ireland, "Preparing for complex interdependent risks: A System of Systems approach to building disaster resilience," Int. J. Disaster Risk Reduct., vol. 9, pp. 181–193, 2014.